Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques

21st International Workshop, APPROX 2018, and
22nd International Workshop, RANDOM 2018
August 20–22, 2018, Princeton, USA

Edited by

Eric Blais
Klaus Jansen
José D. P. Rolim
David Steurer

LIPIcs – Vol. 116 – APPROX/RANDOM 2018 www.dagstuhl.de/lipics

Editors

Eric Blais
University of Waterloo
Waterloo, Canada
eric.blais@uwaterloo.ca

Klaus Jansen
University of Kiel
Kiel, Germany
kj@informatik.uni-kiel.de

Jośe Rolim
University of Geneva
Geneva, Switzerland
Jose.Rolim@unige.chr

David Steurer
Cornell University
New York, USA
dsteurer@cs.cornell.edu

ACM Classification 2012
Mathematics of computing, Theory of computation

ISBN 978-3-95977-085-9

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/978-3-95977-085-9 .

Publication date
August, 2018

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.

Digital Object Identifier: 10.4230/LIPIcs.APPROX-RANDOM.2018.0

ISBN 978-3-95977-085-9 **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.

ISSN 1868-8969

http://www.dagstuhl.de/lipics

Contents

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018).
Editors: Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Contributed Talks of RANDOM

Preface

This volume contains the papers presented at the 21st International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX 2018) and the 22nd International Workshop on Randomization and Computation (RANDOM 2018), which took place concurrently at the at University of Princeton in Princeton, USA during August 20–22, 2018.

APPROX focuses on algorithmic and complexity issues surrounding the development of efficient approximate solutions to computationally difficult problems, and was the 21st in the series after Aalborg (1998), Berkeley (1999), Saarbrücken (2000), Berkeley (2001), Rome (2002), Princeton (2003), Cambridge (2004), Berkeley (2005), Barcelona (2006), Princeton (2007), Boston (2008), Berkeley (2009), Barcelona (2010), Princeton (2011), Boston (2012), Berkeley (2013), Barcelona (2014), Princeton (2015), Paris (2016), and Berkeley (2017). RANDOM is concerned with applications of randomness to computational and combinatorial problems, and was the 22nd workshop in the series following Bologna (1997), Barcelona (1998), Berkeley (1999), Geneva (2000), Berkeley (2001), Harvard (2002), Princeton (2003), Cambridge (2004), Berkeley (2005), Barcelona (2006), Princeton (2007), Boston (2008), Berkeley (2009), Barcelona (2010), Princeton (2011), Boston (2012), Berkeley (2013), Barcelona (2014), Princeton (2015), Paris (2016), and Berkeley (2017).

Topics of interest for APPROX and RANDOM are: approximation algorithms, hardness of approximation, small space, sub-linear time and streaming algorithms, online algorithms, approaches that go beyond worst case analysis, distributed and parallel approximation, embeddings and metric space methods, mathematical programming methods, spectral methods, combinatorial optimization in graphs and networks, algorithmic game theory, mechanism design and economics, computational geometric problems, approximate learning, design and analysis of randomized algorithms, randomized complexity theory, pseudorandomness and derandomization, random combinatorial structures, random walks/Markov chains, expander graphs and randomness extractors, probabilistic proof systems, random projections and embeddings, error-correcting codes, average-case analysis, smoothed analysis, property testing, computational learning theory, and other applications of approximation and randomness.

The volume contains 28 contributed papers, selected by the APPROX Program Committee out of 50 submissions, and 30 contributed papers, selected by the RANDOM Program Committee out of 73 submissions.

We would like to thank all of the authors who submitted papers, the invited speakers, the members of the Program Committees, and the external reviewers. We gratefully acknowledge the Cheriton School of Computer Science of the University of Waterloo in Canada, the Department of Computer Science of the Christian-Albrechts-Universität zu Kiel, the Department of Computer Science of the University of Geneva, and the School of Operations Research and Information Engineering of the Cornell University.

August 2018

Eric Blais, Klaus Jansen
José D. P. Rolim, and David Steurer

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018).
Editors: Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Organization

Program Committees

APPROX 2018

Alina Ene	Boston University, USA
Fabrizio Grandoni	IDSIA, Swiss
Anupam Gupta	Carnegie Mellon University, USA
Pravesh Kothari	Princeton University, USA and the Institute for Advanced Study USA
Lap Chi Lau	University of Waterloo, Canada
Euiwoong Lee	Simons Institute, USA
James Lee	University of Washington, USA
Edo Liberty	Amazon SageMaker, USA
Yury Makarychev	Toyota Technological Institute at Chicago, USA
Neil Olver	Vrije Universiteit Amsterdam, The Netherlands
Yuval Rabani	the Hebrew University of Jerusalem, Israel
Tselil Schramm	Theory of Computation at Harvard, USA
Roy Schwartz	Institute of Technology, Israel
Noah Stephens-Davidowitz	Princeton University, USA
David Steurer (chair)	ETH Zürich, Switzerland
Inbal Talgam-Cohen	Institute of Technology, Israel
Aravindan Vijayaraghavan	Northwestern University, USA
Justin Ward	Queen Mary University of London, United Kingdom
Tony Wirth	University of Melbourne, Australia
Grigory Yaroslavtsev	Indiana University Bloomington, USA

RANDOM 2018

Eric Blais (chair)	University of Waterloo, Canada
Joshua Brody	Swarthmore College, USA
Xi Chen	Columbia University, USA
Gil Cohen	Princeton University, USA and Tel Aviv University, Israel
Andrew Drucker	University of Chicago, USA
Tom Gur	University of California, Berkeley, USA
Daniel Kane	University of California, San Diego, USA
Pravesh Kothari	Princeton University, USA and the Institute for Advanced Study USA
Reut Levi	Weizmann Institute, Israel
Aleksandar Nikolov	University of Toronto, Canada
Lev Reyzin	University of Illinois at Chicago, USA
Noga Ron-Zewi	University of Haifa, Israel
Ron Rosenthal	Technion, Institute of Technology, Israel
Atri Rudra	University at Buffalo, USA
Or Sheffet	University of Alberta, Canada
Thomas Watson	University of Memphis, USA
Yuichi Yoshida	National Institute of Informatics, Japan

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018).
Editors: Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

External Reviewers

Jayadev Acharya
Matthew Anderson
Haris Angelidakis
Sepehr Assadi
Chen Avin
Haim Avron
Pranjal Awasthi
Kyriakos Axiotis
Oren Becker
Shalev Ben-David
Huck Bennett
Mario Berta
Amey Bhangale
Aditya Bhaskara
Sayan Bhattacharya
Vijay Bhattiprolu
Markus Blaeser
Jarosław Błasiok
Greg Bodwin
Martin Böhm
Andrej Bogdanov
Thomas Bosman
Adam Bouland
Clément Canonne
Marco Carmosino
Karthekeyan Chandrasekaran
Eshan Chattopadhyay
Lijie Chen
Xue Chen
Mahdi Cheraghchi
Rajesh Chitnis
Rezaul Chowdhury
Andrea Clementi
Amin Coja-Oghlan
Colin Cooper
Nicolas Crawford
Karthik C. S.
Aparna Das
Syamantak Das
Ilias Diakonikolas
Irit Dinur
Dean Doron
Martin Dyer
Charilaos Efthymiou
Shai Evra
Uriya First
Benjamin Fish
Felix Fischer
Kyle Fox
Zachary Friggstad
Waldo Gálvez
Shashwat Garg
Sumegha Garg
Dmitry Gavinsky
Badih Ghazi
Amin Gohari
Elazar Goldenberg
Sasha Golovnev
Catherine Greenhill
Elena Grigorescu
Ofer Grossman
Anna Gundert
Heng Guo
Guru Guruganesh
Hamed Hassani
Jonathan Hermon
Matty Hoban
William Holland
Samuel Hopkins
Kaave Hosseini
Sihuang Hu
Afrouz Jabalameli
Jafar Jafarov
Mano Vikash Janardhanan
Mani Jayaraman
Madhav Jha
Christos Kalaitzis
Dimitris Kalimeris
Gautam Kamath
Guy Kindler
Hirotada Kobayashi
Laszlo Kozma
Ravishankar Krishnaswamy
Janardhan Kulkarni
Mrinal Kumar
Ron Kupfer
Bundit Laekhanukit
Kevin A. Lai
Amit Levi
Ray Li

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018).
Editors: Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Oren Louidor
Shachar Lovett
Tianren Lu
Matthew Mcpartlon
Moti Medina
Or Meir
Julian Mestre
Sarah Miracle
Dieter Mitsche
Michael Molloy
Morteza Monemizadeh
Ryuhei Mori
Cameron Musco
Christopher Musco
Yonatan Naamad
Vasileios Nakos
Ofer Neiman
Yakov Nekrich
Ryan O'Donnell
Igor Carboni Oliveira
Shayan Oveis Gharan
Denis Pankratov
Yuval Peled
Pan Peng
Richard Peng
Will Perkins
Orestis Plevrakis
Miklos Z. Racz
Md Lutfar Rahman
Ankit Singh Rawat
Daniel Reichman
Benjamin Rossman
Guy Rothblum
Ron Rothblum
Alan Roytman
Rahul Santhanam
Tamas Sarlos
Chris Schwiegelshohn
Kineret Segal
C. Seshadhri
Asaf Shapira
Alexander Sherstov
Nobutaka Shimizu
Igor Shinkar
Sahil Singla
Rene Sitters
Allan Sly
Zhao Song
José A. Soto
Nicholas Spooner
Srikanth Srinivasan
Noah Stephens-Davidowitz
Maxim Sviridenko
Suguru Tamaki
Itzhak Tamo
Li-Yang Tan
Avishay Tal
Roei Tell
Justin Thaler
Madhur Tulsiani
Gyorgy Turan
Rob van Stee
Nithin Varma
Daniel Vaz
Sai Vikneshwar
Emanuele Viola
Ellen Vitercik
Ben Lee Volk
Tal Wagner
Erik Waingarten
Lutz Warnke
Colin White
Andreas Wiese
Ryan Williams
Karl Wimmer
Andrew Wirth
Geoffrey Wolfer
Jinyu Xie
Chihao Zhang
Hongyang Zhang
Yu Zhao
Samson Zhou

List of Authors

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018).
Editors: Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Yonatan Nakar
Vasileios Nakos
Shyam Narayanan

Ryan O'Donnell
Igor C. Oliveira
Izhar Oppenhein

Denis Pankratov

Goonwanth Reddy
Dana Ron
Will Rosenbaum

Kunihiko Sadakane
Rahul Santhanam
Richard Santiago
Kanthi Sarpatwar
Saket Saurabh
Baruch Schieber
Rocco A. Servedio
Hadas Shachnai
F. Bruce Shepherd
Suvrit Sra
Alexandre Stauffer
Daniel Štefankovič
Madhu Sudan
Subhash Suri
Joseph Swernofsky

Li-Yang Tan
Zhihao Gavin Tang
Justin Thaler
Luca Trevisan

Kasturi Varadarajan
Shai Vardi
Daniel Vaz
Rahul Vaze
Michael Viderman
Eric Vigoda
Nisheeth K. Vishnoi

Hao-Ting Wei
Andreas Wiese
Karl Wimmer
David P. Woodruff
Mary Wootters
Xiaowei Wu
Angela Wuu

Chaoping Xing

Kuan Yang
Ozan Yıldız
Yuichi Yoshida
Chen Yuan

Meirav Zehavi
Jiapeng Zhang
Yuhao Zhang
Yu Zhao
Samson Zhou

Polylogarithmic Approximation Algorithms for Weighted-$\mathcal{F}$-Deletion Problems

Akanksha Agrawal
Institute for Computer Science and Control, Hungarian Academy of Sciences (MTA SZTAKI), Budapest, Hungary
agrawal.akanksha@sztaki.mta.hu

Daniel Lokshtanov
University of Bergen, Norway
daniello@ii.uib.no

Pranabendu Misra
University of Bergen, Norway
pranabendu.misra@ii.uib.no

Saket Saurabh
Institute of Mathematical Sciences, HBNI, Chennai, India,
University of Bergen, Norway, and UMI ReLax
saket@imsc.res.in

Meirav Zehavi
Ben-Gurion University, Beersheba, Israel
meiravze@bgu.ac.il

Abstract

Let $\mathcal{F}$ be a family of graphs. A canonical vertex deletion problem corresponding to $\mathcal{F}$ is defined as follows: given an n-vertex undirected graph G and a weight function $w : V(G) \to \mathbb{R}^+$, find a minimum weight subset $S \subseteq V(G)$ such that $G - S$ belongs to $\mathcal{F}$. This is known as Weighted $\mathcal{F}$ Vertex Deletion problem. In this paper we devise a recursive scheme to obtain $\mathcal{O}(\log^{\mathcal{O}(1)} n)$-approximation algorithms for such problems, building upon the classical technique of finding *balanced separators* in a graph. Roughly speaking, our scheme applies to those problems, where an optimum solution S together with a well-structured set X, form a balanced separator of the input graph. In this paper, we obtain the first $\mathcal{O}(\log^{\mathcal{O}(1)} n)$-approximation algorithms for the following vertex deletion problems.

- Let $\mathcal{F}$ be a finite set of graphs containing a planar graph, and $\mathcal{F} = \mathcal{G}(\mathcal{F})$ be the family of graphs such that every graph $H \in \mathcal{G}(\mathcal{F})$ excludes all graphs in $\mathcal{F}$ as minors. The vertex deletion problem corresponding to $\mathcal{F} = \mathcal{G}(\mathcal{F})$ is the Weighted Planar $\mathcal{F}$-Minor-Free Deletion (WP$\mathcal{F}$-MFD) problem. We give randomized and deterministic approximation algorithms for WP$\mathcal{F}$-MFD with ratios $\mathcal{O}(\log^{1.5} n)$ and $\mathcal{O}(\log^2 n)$, respectively. Previously, only a randomized constant factor approximation algorithm for the *unweighted* version of the problem was known [FOCS 2012].
- We give an $\mathcal{O}(\log^2 n)$-factor approximation algorithm for Weighted Chordal Vertex Deletion (WCVD), the vertex deletion problem to the family of chordal graphs. On the way to this algorithm, we also obtain a constant factor approximation algorithm for Multicut on chordal graphs.
- We give an $\mathcal{O}(\log^3 n)$-factor approximation algorithm for Weighted Distance Hereditary Vertex Deletion (WDHVD), also known as Weighted Rankwidth-1 Vertex Deletion (WR-1VD). This is the vertex deletion problem to the family of distance hereditary graphs, or equivalently, the family of graphs of rankwidth one.

We believe that our recursive scheme can be applied to obtain $\mathcal{O}(\log^{\mathcal{O}(1)} n)$-approximation algorithms for many other problems as well.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018).
Editors: Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer; Article No. 1; pp. 1:1–1:15

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

2012 ACM Subject Classification Mathematics of computing → Approximation algorithms

Keywords and phrases Approximation Algorithms, Planar-$\mathcal{F}$-Deletion, Separator

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.1

Related Version A full version of the paper is available at [2], `https://arxiv.org/abs/1707.04908`.

Funding This research has received funding from the European Research Council under ERC grant no. 306992 PARAPPROX, ERC grant no. 715744 PaPaALG, and ERC grant no. 725978 SYSTEMATICGRAPH.

Acknowledgements We sincerely thank Nikhil Bansal and Seeun William Umboh for explaining their paper to us, and for several discussions on Weighted Planar $\mathcal{F}$-Minor-Free Deletion.

1 Introduction

Let $\mathcal{F}$ be a family of undirected graphs. Then a natural optimization problem is as follows.

Weighted $\mathcal{F}$ Vertex Deletion
Input: An undirected graph G and a weight function $w : V(G) \to \mathbb{R}^+$.
Question: Find a minimum weight subset $S \subseteq V(G)$ such that $G - S$ belongs to $\mathcal{F}$.

The Weighted $\mathcal{F}$ Vertex Deletion problem captures a wide class of node (or vertex) deletion problems that have been studied from the 1970s. For example, when $\mathcal{F}$ is the family of independent sets, forests, bipartite graphs, planar graphs, and chordal graphs, then the corresponding vertex deletion problem corresponds to Weighted Vertex Cover, Weighted Feedback Vertex Set, Weighted Vertex Bipartization (also called Weighted Odd Cycle Transversal), Weighted Planar Vertex Deletion and Weighted Chordal Vertex Deletion, respectively. By a classic theorem of Lewis and Yannakakis [29], the decision version of the Weighted $\mathcal{F}$ Vertex Deletion problem – deciding whether there exists a set S weight at most k, such that removing S from G results in a graph with property Π – is NP-complete for every non-trivial hereditary property[1] Π.

Characterizing the graph properties, for which the corresponding vertex deletion problems can be approximated within a bounded factor in polynomial time, is a long standing open problem in approximation algorithms [43]. In spite of a long history of research, we are still far from a complete characterization. Constant factor approximation algorithms for Weighted Vertex Cover are known since 1970s [5, 32]. Lund and Yannakakis observed that the vertex deletion problem for any hereditary property with a "finite number of minimal forbidden induced subgraphs" can be approximated within a constant ratio [30]. They conjectured that for every nontrivial, hereditary property Π with an infinite forbidden set, the corresponding vertex deletion problem cannot be approximated within a constant ratio. However, it was later shown that Weighted Feedback Vertex Set, which doesn't have a finite forbidden set, admits a constant factor approximation [3, 6], thus disproving their conjecture. On the

[1] A graph property Π is simply a family of graphs closed under isomorphism, and it is called *non-trivial* if there exists an infinite number of graphs that are in Π, as well as an infinite number of graphs that are not in Π. A non-trivial graph property Π is called *hereditary* if $G \in \Pi$ implies that every induced subgraph of G is also in Π.

other hand a result by Yannakakis [42] shows that, for a wide range of graph properties Π, approximating the minimum number of vertices to delete in order to obtain a *connected* graph with the property Π within a factor $n^{1-\varepsilon}$ is NP-hard. We refer to [42] for the precise list of graph properties to which this result applies to, but it is worth mentioning the list includes the class of acyclic graphs and the class of outerplanar graphs.

In this paper, we explore the approximability of Weighted $\mathcal{F}$ Vertex Deletion for several different families $\mathcal{F}$ and design $\mathcal{O}(\log^{\mathcal{O}(1)} n)$-factor approximation algorithms for these problems. More precisely, our results are as follows.

1. Let $\mathcal{F}$ be a finite set of graphs that includes a planar graph. Let $\mathcal{F} = \mathcal{G}(\mathcal{F})$ be the family of graphs such that every graph $H \in \mathcal{G}(\mathcal{F})$ does not contain a graph from $\mathcal{F}$ as a minor. The vertex deletion problem corresponding to $\mathcal{F} = \mathcal{G}(\mathcal{F})$ is known as the Weighted Planar $\mathcal{F}$-Minor-Free Deletion (WP$\mathcal{F}$-MFD). The WP$\mathcal{F}$-MFD problem is a very generic problem and by selecting different sets of forbidden minors $\mathcal{F}$, one can obtain various fundamental problems such as Weighted Vertex Cover, Weighted Feedback Vertex Set or Weighted Treewidth η-Deletion. Our first result is a randomized $\mathcal{O}(\log^{1.5} n)$-factor (deterministic $\mathcal{O}(\log^2 n)$-factor) approximation algorithm for WP$\mathcal{F}$-MFD, for any finite $\mathcal{F}$ that contains a planar graph.
2. We give an $\mathcal{O}(\log^2 n)$-factor approximation algorithm for Weighted Chordal Vertex Deletion (WCVD), the vertex deletion problem corresponding to the family of chordal graphs. On the way to this algorithm, we also obtain a constant factor approximation algorithm for Weighted Multicut in chordal graphs.
3. We give an $\mathcal{O}(\log^3 n)$-factor approximation algorithm for Weighted Distance Hereditary Vertex Deletion (WDHVD), also known as the Weighted Rankwidth-1 Vertex Deletion (WR-1VD) problem. It is the vertex deletion problem corresponding to the family of distance hereditary graphs, or equivalently graphs of rankwidth 1.

All our algorithms follow the same recursive scheme: find "well structured balanced separators" in the graph by exploiting the properties of the family $\mathcal{F}$, and then use structure of the balanced separator to obtain a approximate solution. In the following, we first describe the methodology by which we design all these approximation algorithms. Then, we give a brief overview, consisting of known results and our contributions, for each problem we study. Let us also mention that these problems inherit the hardness of approximation of Vertex Cover via simple reductions. In particular, they don't admit a PTAS (polynomial time approximation scheme) unless P = NP.

Our Methods

Multicommodity max-flow min-cut theorems are a classical technique in designing approximation algorithms, which was pioneered by Leighton and Rao in their seminal paper [28]. This approach can be viewed as using balanced vertex (or edge) separators[2] in a graph to obtain a divide-and-conquer approximation algorithm. In a typical application, the optimum solution S, forms a balanced separator of the graph. Thus, the idea is to find a minimum cost balanced separator W of the graph and add it to the solution, and then recursively solve the problem on each of the connected components. This leads to an $\mathcal{O}(\log^{\mathcal{O}(1)} n)$-factor approximation algorithm for the problem in question.

[2] Roughly speaking, a *balanced vertex separator* is a set of vertices W, such that any connected component of $G - W$ contains at most $\frac{2}{3}$ of the vertices of G.

Our recursive scheme is a strengthening of this approach which exploits the structural properties of the family $\mathcal{F}$. Here the optimum solution S^* need not be a balanced separator of the graph. Indeed, a balanced separator of the graph could be much larger than S^*. Rather, S^* along with a possibly large but well-structured subset of vertices X, forms a balanced separator of the graph. We then exploit the presence of such a balanced separator in the graph to compute an approximate solution. Consider a family $\mathcal{F}$ for which Weighted $\mathcal{F}$ Vertex Deletion is amenable to our approach, and let G be an instance of this problem. Let S be the approximate solution that we will compute. Our approximation algorithm has the following steps:

1. Find a well-structured set X, such that $G - X$ has a balanced separator W which is not too costly.
2. Next, compute the balanced separator W of $G - X$ using the known factor $\mathcal{O}(\sqrt{\log n})$-approximation algorithm (or deterministic $\mathcal{O}(\log n)$-approximation algorithm) for Weighted Vertex Separators [12, 28]. Then add W into the solution set S and recursively solve the problem on each connected component of $G-(X\cup S)$. Let $S_1, \cdots, S_\ell$ be the solutions returned by the recursive calls. We add $S_1, \cdots, S_\ell$ to the solution S.
3. Finally, we add X back into the graph and consider the instance $(G - S) \cup X$. Observe that, $V(G - S)$ can be partitioned into $V' \uplus X$, where $G[V']$ belongs to $\mathcal{F}$ and X is a well-structured set. We call such instances, the *special case* of Weighted $\mathcal{F}$ Vertex Deletion. We apply an approximation algorithm that exploits the structural properties of the special case to compute a solution.

Now consider the problem of finding the structure X. One way is to enumerate all the candidates for X and then pick the one where $G - X$ has a balanced vertex separator of least cost – this separator plays the role of W. However, the number of candidates for X in a graph could be too many to enumerate in polynomial time. For example, in the case of Weighted Chordal Vertex Deletion, the set X will be a clique in the graph, and the number of maximal cliques in a graph on n vertices could be as many as $3^{\frac{n}{3}}$ [31]. Hence, we cannot enumerate and test every candidate structure in polynomial time. However, we can exploit certain structural properties of family $\mathcal{F}$, to reduce the number of candidates for X in the graph. In our problems, we "tidy up" the graph by removing "short obstructions" that forbid the graph from belonging to the family $\mathcal{F}$. Then one can obtain an upper bound on the number of candidate structures. In the above example, recall that a graph G is chordal if and only if there are no induced cycles of length 4 or more. It is known that a graph G without any induced cycle of length 4 has at most $\mathcal{O}(n^2)$ maximal cliques [11]. Observe that, we can greedily compute a set of vertices which intersects all induced cycles of length 4 in the graph. Therefore, at the cost of factor 4 in the approximation ratio, we can ensure that the graph has only polynomially many maximal cliques. Hence, one can enumerate all maximal cliques in the remaining graph [41] to test for X.

Next consider the task of solving an instance of the special case of the problem. We again apply a recursive scheme, but now with the advantage of a much more structured graph. By a careful modification of an LP solution to the instance, we eventually reduce it to instances of Weighted Multicut. In the above example, for Weighted Chordal Vertex Deletion we obtain instances of Weighted Multicut on a chordal graph. We follow this approach for all three problems that we study in this paper. We believe our recursive scheme can be applied to obtain $\mathcal{O}(\log^{\mathcal{O}(1)} n)$-approximation algorithms for Weighted $\mathcal{F}$ Vertex (Edge) Deletion corresponding to several other graph families $\mathcal{F}$.

Weighted Planar $\mathcal{F}$-Minor-Free Deletion. Let $\mathcal{F}$ be a finite set of graphs containing a planar graph. The vertex deletion problem corresponding to $\mathcal{F}$ is defined as follows.

Weighted Planar $\mathcal{F}$-Minor-Free Deletion (WP$\mathcal{F}$-MFD)
Input: An undirected graph G and a weight function $w : V(G) \to \mathbb{R}^+$.
Question: Find a minimum weight subset $S \subseteq V(G)$ such that $G - S$ does not contain any graph in $\mathcal{F}$ as a minor.

The WP$\mathcal{F}$-MFD problem is a very generic problem that encompasses several known problems. To explain the versatility of the problem, we require a few definitions. A graph H is called a *minor* of a graph G if we can obtain H from G by a sequence of vertex deletions, edge deletions and edge contractions, and a family of graphs $\mathcal{F}$ is called *minor closed* if $G \in \mathcal{F}$ implies that every minor of G is also in $\mathcal{F}$. Given a graph family $\mathcal{F}$, by $\mathsf{ForbidMinor}(\mathcal{F})$ we denote the family of graphs such that $G \in \mathcal{F}$ if and only if G does not contain any graph in $\mathsf{ForbidMinor}(\mathcal{F})$ as a minor. By the celebrated Graph Minor Theorem of Robertson and Seymour, every minor closed family $\mathcal{F}$ is characterized by a finite family of forbidden minors [39]. That is, $\mathsf{ForbidMinor}(\mathcal{F})$ has finite size. Indeed, the size of $\mathsf{ForbidMinor}(\mathcal{F})$ depends on the family $\mathcal{F}$. Now for a finite collection of graphs $\mathcal{F}$, as above, we may define the Weighted $\mathcal{F}$-Minor-Free Deletion problem. And observe that, even though the definition of Weighted $\mathcal{F}$-Minor-Free Deletion we only consider finite sized $\mathcal{F}$, this problem actually encompasses deletion to every minor closed family of graphs. Let $\mathcal{G}$ be the set of all finite undirected graphs, and let $\mathcal{L}$ be the family of all finite subsets of $\mathcal{G}$. Thus, every element $\mathcal{F} \in \mathcal{L}$ is a finite set of graphs, and throughout the paper we assume that $\mathcal{F}$ is explicitly given. In this paper, we show that when $\mathcal{F} \in \mathcal{L}$ contains at least one planar graph, then it is possible to obtain an $\mathcal{O}(\log^{\mathcal{O}(1)} n)$-factor approximation algorithm for WP$\mathcal{F}$-MFD.

The case where $\mathcal{F}$ contains a planar graph, while being considerably more restricted than the general case, already encompasses a number of the well-studied instances of WP$\mathcal{F}$-MFD. For example, when $\mathcal{F} = \{K_2\}$, a complete graph on two vertices, this is the Weighted Vertex Cover problem. When $\mathcal{F} = \{C_3\}$, a cycle on three vertices, this is the Weighted Feedback Vertex Set problem. Another fundamental problem, which is also a special case of WP$\mathcal{F}$-MFD, is Weighted Treewidth-η Vertex Deletion or Weighted η-Transversal. Here the task is to delete a minimum weight vertex subset to obtain a graph of treewidth at most η. Since any graph of treewidth η excludes a $(\eta + 1) \times (\eta + 1)$ grid as a minor, we have that the set $\mathcal{F}$ of forbidden minors of treewidth η graphs contains a planar graph. Treewidth-η Vertex Deletion plays an important role in generic efficient polynomial time approximation schemes based on Bidimensionality theory [16, 17]. Other examples of Planar $\mathcal{F}$-Minor-Free Deletion problems that can be found in the literature on approximation and parameterized algorithms, are the cases of $\mathcal{F}$ being $\{K_{2,3}, K_4\}$, $\{K_4\}$, $\{\theta_c\}$, and $\{K_3, T_2\}$, which correspond to removing vertices to obtain an outerplanar graph, a series-parallel graph, a diamond graph, and a graph of pathwidth 1, respectively.

Apart from the case of Weighted Vertex Cover [5, 32] and Weighted Feedback Vertex Set [3, 6], there was not much progress on approximability/non-approximability of WP$\mathcal{F}$-MFD until the work of Fiorini, Joret, and Pietropaoli [13], which gave a constant factor approximation algorithm for the case of WP$\mathcal{F}$-MFD where $\mathcal{F}$ is a diamond graph, i.e., a graph with two vertices and three parallel edges. In 2011, Fomin et al. [14] considered Planar $\mathcal{F}$-Minor-Free Deletion (i.e. the unweighted version of WP$\mathcal{F}$-MFD) in full generality and designed a randomized (deterministic) $\mathcal{O}(\log^{1.5} n)$-factor ($\mathcal{O}(\log^2 n)$-factor) approximation algorithm for it. Later, Fomin et al. [15] gave a randomized constant factor approximation algorithm for Planar $\mathcal{F}$-Minor-Free Deletion. Our algorithm for WP$\mathcal{F}$-MFD extends this result to the weighted setting, at the cost of increasing the approximation factor to $\log^{\mathcal{O}(1)} n$.

▶ **Theorem 1.** *For every set $\mathcal{F} \in \mathcal{L}$,* WP$\mathcal{F}$-MFD *admits a randomized (deterministic) $\mathcal{O}(\log^{1.5} n)$-factor ($\mathcal{O}(\log^2 n)$-factor) approximation algorithm.*

We mention some recent related works. Bansal et al. [4] have studied the edge deletion version of the Treewidth-η Vertex Deletion problem, under the name Bounded Treewidth Interdiction Problem, and gave a bicriteria approximation algorithm. In particular, for a graph G and an integer $\eta > 0$, they gave a polynomial time algorithm that finds a subset of edges F' of G such that $|F'| \leq \mathcal{O}((\log n \log\log n) \cdot \mathsf{opt})$ and the treewidth of $G - F'$ is $\mathcal{O}(\eta \log \eta)$. In our setting where η is a fixed constant, this immediately implies a factor $\mathcal{O}(\log n \log\log n)$ approximation algorithm for the edge deletion version of WP$\mathcal{F}$-MFD.[3] However, it is not immediately clear if their approach can be extended to WP$\mathcal{F}$-MFD.[4] Very recently, Gupta et al. [22] have given $\mathcal{O}(\log \ell)$ approximation algorithm for (unweighted) Planar $\mathcal{F}$-Minor-Free Deletion, where ℓ is the maximum number of vertices in any planar graph in $\mathcal{F}$.

Weighted Chordal Vertex Deletion. This problem is defined as follows.

Weighted Chordal Vertex Deletion (WCVD)
Input: An undirected graph G and a weight function $w : V(G) \to \mathbb{R}^+$.
Question: Find a minimum weight subset $S \subseteq V(G)$ such that $G - S$ is a chordal graph.

The class of chordal graphs is a natural class of graphs that has been extensively studied from the viewpoints of Graph Theory and Algorithm Design. Many important problems that are NP-hard on general graphs, such as Independent Set, and Graph Coloring are solvable in polynomial time once restricted to the class of chordal graphs [21]. Recall that a graph is chordal if and only if it does not have any induced cycle of length 4 or more. Thus, Chordal Vertex Deletion (CVD) can be viewed as a natural variant of the classic Feedback Vertex Set (FVS). Indeed, while the objective of FVS is to eliminate all cycles, the CVD problem only asks us to eliminate induced cycles of length 4 or more. Despite the apparent similarity between the objectives of these two problems, the design of approximation algorithms for WCVD is very challenging. In particular, chordal graphs can be dense – indeed, a clique is a chordal graph. As we cannot rely on the sparsity of output, our approach must deviate from those employed by approximation algorithms from FVS. That being said, chordal graphs still retain some properties that resemble those of trees, and these properties are utilized by our algorithm. Prior to our work, only two non-trivial approximation algorithms for CVD were known. The first one, by Jansen and Pilipczuk [26], is a deterministic $\mathcal{O}(\mathsf{opt}^2 \log \mathsf{opt} \log n)$-factor approximation algorithm, and the second one, by Agrawal et al. [1], is a deterministic $\mathcal{O}(\mathsf{opt} \log^2 n)$-factor approximation algorithm. The second result implies that CVD admits an $\mathcal{O}(\sqrt{n} \log n)$-factor approximation algorithm.[5] In this paper we obtain the first $\mathcal{O}(\log^{\mathcal{O}(1)} n)$-approximation algorithm for WCVD.

▶ **Theorem 2.** CVD *admits a deterministic $\mathcal{O}(\log^2 n)$-factor approximation algorithm.*

---

[3] One can run their algorithm first and remove the solution output by their algorithm to obtain a graph of treewidth at most $\mathcal{O}(\eta \log \eta)$. Then one can find an optimal solution using standard dynamic programming.

[4] We thank Nikhil Bansal and Seeun William Umboh for several discussions and for pointing us that their algorithm does not work for WP$\mathcal{F}$-MFD.

[5] If $\mathsf{opt} \geq \sqrt{n}/\log n$, we output a greedy solution to the input graph, and otherwise we have that $\mathsf{opt} \log^2 n \leq \sqrt{n} \log n$, hence we call the $\mathcal{O}(\mathsf{opt} \log^2 n)$-factor approximation algorithm.

While this approximation algorithm follows our general scheme, it also requires us to incorporate several new ideas. In particular, to implement the third step of the scheme, we need to design a different $\mathcal{O}(\log n)$-factor approximation algorithm for the special case of WCVD where the vertex-set of the input graph G can be partitioned into two sets, X and $V(G) \setminus X$, such that $G[X]$ is a clique and $G[V(G) \setminus X]$ is a chordal graph. This approximation algorithm is again based on recursion, but it is more involved. At each recursive call, it carefully manipulates a fractional solution of a special form. Moreover, to ensure that its current problem instance is divided into two subinstances that are independent and simpler than their origin, we introduce multicut constraints. In addition to these constraints, we keep track of the complexity of the subinstances, which is measured via the cardinality of the maximum independent set in the graph. Our multicut constraints result in an instance of Weighted Multicut, which we ensure is on a chordal graph.

Weighted Multicut
Input: An undirected graph G, a weight function $w : V(G) \to \mathbb{R}^+$ and a set $T = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ of k pairs of vertices of G.
Question: Find a minimum weight subset $S \subseteq V(G)$ such that for any pair $(s_i, t_i) \in \mathcal{T}$, $G - S$ does not have any path between s_i and t_i.

For Weighted Multicut on chordal graphs, no constant-factor approximation algorithm was previously known. We remark that Weighted Multicut is NP-hard on trees [19], and hence it is also NP-hard on chordal graphs. We design the first such algorithm, which our main algorithm employs as a black box.

▶ **Theorem 3.** Weighted Multicut *admits a constant-factor approximation algorithm on chordal graphs.*

This algorithm is inspired by the work of Garg, Vazirani and Yannakakis on Weighted Multicut on trees [19]. Here, we carefully exploit the well-known characterization of the class of chordal graphs as the class of graphs that admit clique forests. We believe that this result is of independent interest. The algorithm by Garg, Vazirani and Yannakakis [19] is a classic primal-dual algorithm. A more recent algorithm, by Golovin, Nagarajan and Singh [20], uses total unimodularity to obtain a different algorithm for Multicut on trees.

Weighted Distance Hereditary Vertex Deletion. Let us start with the formal definition.

Weighted Distance Hereditary Vertex Deletion (WDHVD)
Input: An undirected graph G and a weight function $w : V(G) \to \mathbb{R}^+$.
Question: Find a minimum weight subset $S \subseteq V(G)$ such that $G - S$ is a distance hereditary graph.

A graph G is a *distance hereditary graph* (also called a completely separable graph [23]) if the distances between vertices in every connected induced subgraph of G are the same as in the graph G. Distance hereditary graphs were named and first studied by Hworka [25]. However, an equivalent family of graphs was earlier studied by Olaru and Sachs [40] and shown to be perfect. It was later discovered that these graphs are precisely the graphs of rankwidth one [33].

Rankwidth is a graph parameter introduced by Oum and Seymour [36] to approximate yet another graph parameter called Cliquewidth. The notion of cliquewidth was defined by Courcelle and Olariu [9] as a measure of how "clique-like" the input graph is. This is similar to the notion of treewidth, which measures how "tree-like" the input graph is. One of the

main motivations was that several NP-complete problems become tractable on the family of cliques (complete graphs), the assumption was that these algorithmic properties extend to "clique-like" graphs [8]. However, computing cliquewidth and the corresponding cliquewidth decomposition seems to be computationally intractable. This then motivated the notion of rankwidth, which is a graph parameter that approximates cliquewidth well while also being algorithmically tractable [36, 34]. For more information on cliquewidth and rankwidth, we refer to the surveys by Hlinený et al. [24] and Oum [35].

As algorithms for Treewidth-η Vertex Deletion are applied as subroutines to solve many graph problems, we believe that algorithms for Weighted Rankwidth-η Vertex Deletion (WR-ηVD) will be useful in this respect. In particular, Treewidth-η Vertex Deletion has been considered in designing efficient approximation, kernelization and fixed parameter tractable algorithms for WP$\mathcal{F}$-MFD and its unweighted counterpart Planar $\mathcal{F}$-Minor-Free Deletion [4, 14, 16, 17, 18]. Along similar lines, we believe that WR-ηVD and its unweighted counterpart will be useful in designing efficient approximation, kernelization and fixed parameter tractable algorithms for Weighted $\mathcal{F}$ Vertex Deletion where $\mathcal{F}$ is characterized by a finite family of forbidden *vertex minors* [33].

Recently, Kim and Kwon [27] designed an $\mathcal{O}(\mathsf{opt}^2 \log n)$-factor approximation algorithm for Distance Hereditary Vertex Deletion (DHVD). This result implies that DHVD admits an $\mathcal{O}(n^{2/3} \log n)$-factor approximation algorithm. In this paper, we take first step towards obtaining a good approximation algorithm for WR-ηVD by designing a $\mathcal{O}(\log^{\mathcal{O}(1)} n)$-factor approximation algorithm for WDHVD.

▶ **Theorem 4.** WDHVD *or* WR-1VD *admits an* $\mathcal{O}(\log^3 n)$*-factor approximation algorithm.*

We note that several steps of our approximation algorithm for WR-1VD can be generalized for an approximation algorithm for WR-ηVD and thus we believe that our approach should yield an $\mathcal{O}(\log^{\mathcal{O}(1)} n)$-factor approximation algorithm for WR-ηVD. We leave that as an interesting open problem for the future.

Organization of the paper

Due to space constraints, we only present the details of Weighted Planar $\mathcal{F}$-Minor-Free Deletion in this extended abstract. The details of the algorithms for Weighted Chordal Vertex Deletion and Weighted Distance Hereditary Vertex Deletion will appear in the full version of the paper (see [2]). Graph theoretic preliminaries have been deferred to the appendix.

2 Approximation Algorithm for WP$\mathcal{F}$-MFD

In this section we prove Theorem 1. We can assume that the weight $w(v)$ of each vertex $v \in V(G)$ is positive, else we can insert v into any solution. Below we state a result from [37], which will be useful in our algorithm.

▶ **Proposition 5** ([37])**.** *Let $\mathcal{F}$ be a finite set of graphs such that $\mathcal{F}$ contains a planar graph. Then, any graph G that excludes any graph from $\mathcal{F}$ as a minor satisfies* $\mathsf{tw}(G) \leq c = c(\mathcal{F})$.

We let $c = c(\mathcal{F})$ to be the constant returned by Proposition 5. The approximation algorithm for WP$\mathcal{F}$-MFD comprises of two components. The first component handles the special case where the vertex set of input graph G can be partitioned into two sets C and X such that $|C| \leq c+1$ and $H = G[X]$ is an $\mathcal{F}$-minor free graph. We note that there can be edges between vertices in C and vertices in H. We show that for these special instances, in polynomial time we can compute the size of the optimum solution and a set realizing it.

The second component is a recursive algorithm that solves general instances of the problem. Here, we gradually disintegrate the general instance until it becomes an instance of the special type where we can resolve it in polynomial time. More precisely, for each guess of $c+1$ sized subgraph M of G, we find a small separator S (using an approximation algorithm) that together with M breaks the input graph into two graphs significantly smaller than their origin. It first removes $M \cup S$, and solves each of the two resulting subinstances by calling itself recursively; then, it inserts M back into the graph, and uses the solutions it obtained from the recursive calls to construct an instance of the special case which is then solved by the first component.

2.1 Constant sized graph + $\mathcal{F}$-minor free graph

We first handle the special case where the input graph G consists of a graph M of size at most $c+1$ and an $\mathcal{F}$-minor free graph H. We refer to this algorithm as Special-WP. More precisely, along with the input graph G and the weight function w, we are also given a graph M with at most $c+1$ vertices and an $\mathcal{F}$-minor free graph H such that $V(G) = V(M) \cup V(H)$, where the vertex-sets $V(M)$ and $V(H)$ are disjoint. Note that the edge-set $E(G)$ may contain edges between vertices in M and vertices in H. We will show that such instances may be solved optimally in polynomial time. We start with the following easy observation.

▶ **Observation 6.** *Let G be a graph with $V(G) = X \uplus Y$, such that $|X| \leq c+1$ and $G[Y]$ is an $\mathcal{F}$-minor free graph. Then, the treewidth of G is at most $2c+1$.*

▶ **Lemma 7.** *Let G be a graph of treewidth t with a non-negative weight function w on the vertices, and let $\mathcal{F}$ be a finite family of graphs. Then, one can compute a minimum weight vertex set S such that $G - S$ is $\mathcal{F}$-minor free, in time $f(q,t) \cdot n$, where n is the number of vertices in G and q is a constant that depends only on $\mathcal{F}$.*

Proof. This follows from the fact that finding such a set S is expressible as an MSO-optimization formula ϕ whose length, q, depends only on the family $\mathcal{F}$ [15]. Then, by Theorem 7 [7], we can compute an optimal sized set S in time $f(q,t) \cdot n$. ◀

Now, we apply the above lemma to the graph G and the family $\mathcal{F}$, and obtain a minimum weight set S such that $G - S$ is $\mathcal{F}$-minor free.

2.2 General Graphs

We proceed to handle general instances by developing a $d \cdot \log^2 n$-factor approximation algorithm for WP$\mathcal{F}$-MFD, Gen-WP-APPROX, thus proving the correctness of Theorem 1. The exact value of the constant d will be determined later.

Recursion. We define each call to our algorithm Gen-WP-APPROX to be of the form (G', w'), where (G', w') is an instance of WP$\mathcal{F}$-MFD such that G' is an induced subgraph of G, and we denote $n' = |V(G')|$.

Goal. For each recursive call Gen-WP-APPROX(G', w'), we aim to prove the following.

▶ **Lemma 8.** Gen-WP-APPROX *returns a solution that is at least* opt *and at most* $\frac{d}{2} \cdot \log^2 n' \cdot$ opt. *Moreover, it returns a subset $U \subseteq V(G')$ that realizes the solution.*

At each recursive call, the size of the graph G' becomes smaller. Thus, when we prove that Lemma 8 is true for the current call, we assume that the approximation factor is bounded by $\frac{d}{2} \cdot \log^2 \widehat{n} \cdot \mathsf{opt}$ for any call where the size $\widehat{n}$ of the vertex-set of its graph is strictly smaller than n'.

Termination. In polynomial time we can test whether G' has a minor $F \in \mathcal{F}$ [38]. Furthermore, for each $M \subseteq V(G)$ of size at most $c+1$, we can check if $G - M$ has a minor $F \in \mathcal{F}$. If $G - M$ is $\mathcal{F}$-minor free then we are in a special instance, where $G - M$ is $\mathcal{F}$ minor free and M is a constant sized graph. We optimally resolve this instance in polynomial time using the algorithm Special-WP. Since we output an optimal sized solution in the base cases, we thus ensure that at the base case of our induction Lemma 8 holds.

Recursive Call. For the analysis of a recursive call, let S^* denote a hypothetical set that realizes the optimal solution opt of the current instance (G', w'). Let (F, β) be a forest decomposition of $G' - S^*$ of width at most c, whose existence is guaranteed by Proposition 5. Using standard arguments on forests we have the following observation.

▶ **Observation 9.** *There exists a node $v \in V(F)$ such that $\beta(v)$ is a balanced separator for $G' - S^*$.*

From Observation 9 we know that there exists a node $v \in V(F)$ such that $\beta(v)$ is a balanced separator for $G' - S^*$. This together with the fact that $G' - S^*$ has treewidth at most c results in the following observation.

▶ **Observation 10.** *There exist a subset $M \subseteq V(G')$ of size at most $c+1$ and a subset $S \subseteq V(G') \setminus M$ of weight at most opt such that $M \cup S$ is a balanced separator for G'.*

This gives us a polynomial time algorithm as stated in the following lemma.

▶ **Lemma 11.** *There is a deterministic (randomized) algorithm which in polynomial-time finds $M \subseteq V(G')$ of size at most $c+1$ and a subset $S \subseteq V(G') \setminus M$ of weight at most $q \cdot \log n' \cdot \mathsf{opt}$ ($q^* \cdot \sqrt{\log n'} \cdot \mathsf{opt}$) for some fixed constant q (q^*) such that $M \cup S$ is a balanced separator for G'.*

Proof. Note that we can enumerate every $M \subseteq V(G')$ of size at most $c+1$ in time $\mathcal{O}(n^c)$. For each such M, we can either run the randomized $q^* \cdot \sqrt{\log n'}$-factor approximation algorithm by Feige et al. [12] or the deterministic $q \cdot \log n'$-factor approximation algorithm by Leighton and Rao [28] to find a balanced separator S_M of $G' - M$. Here, q and q^* are fixed constants. By Observation 10, there is a set S in $\{S_M : M \subseteq V(G') \text{ and } M \leq c+1\}$ such that $w(S) \leq q \cdot \log n' \cdot \mathsf{opt}$ ($w(S) \leq q^* \cdot \sqrt{\log n'} \cdot \mathsf{opt}$). Thus, the desired output is a pair (M, S) where M is one of the vertex subset of size at most $c+1$ such that $S_M = S$. ◀

We call the algorithm in Lemma 11 to obtain a pair (M, S). Since $M \cup S$ is a balanced separator for G', we can partition the set of connected components of $G' - (M \cup S)$ into two sets, $\mathcal{A}_1$ and $\mathcal{A}_2$, such that for $V_1 = \bigcup_{A \in \mathcal{A}_1} V(A)$ and $V_2 = \bigcup_{A \in \mathcal{A}_2} V(A)$ it holds that $n_1, n_2 \leq \frac{2}{3} n'$ where $n_1 = |V_1|$ and $n_2 = |V_2|$. We remark that we use different algorithms for finding a balanced separator in Lemma 11 based on whether we are looking for a randomized algorithm or a deterministic algorithm.

Next, we define two inputs of (the general case of) WP$\mathcal{F}$-MFD: $I_1 = (G'[V_1], w'|_{V_1})$ and $I_2 = (G'[V_2], w'|_{V_2})$. Let opt_1 and opt_2 denote the optimal solutions to I_1 and I_2, respectively. Observe that since $V_1 \cap V_2 = \emptyset$, it holds that $\mathsf{opt}_1 + \mathsf{opt}_2 \leq \mathsf{opt}$. We solve each of the

subinstances by recursively calling algorithm Gen-WP-APPROX. By the inductive hypothesis, we thus obtain two sets, S_1 and S_2, such that $G'[V_1] - S_1$ and $G'[V_2] - S_2$ are $\mathcal{F}$-minor free graphs, and $w'(S_1) \leq \frac{d}{2} \cdot \log^2 n_1 \cdot \mathsf{opt}_1$ and $w'(S_2) \leq \frac{d}{2} \cdot \log^2 n_2 \cdot \mathsf{opt}_2$.

We proceed by defining an input of the special case of WP$\mathcal{F}$-MFD: $J = (G'[(V_1 \cup V_2 \cup M) \setminus (S_1 \cup S_2)], w'|_{(V_1 \cup V_2 \cup M) \setminus (S_1 \cup S_2)})$. Observe that $G'[V_1 \setminus S_1]$ and $G'[V_2 \setminus S_2]$ are $\mathcal{F}$-minor free graphs and there are no edges between vertices in V_1 and vertices in V_2 in $G' - M$, and M is of constant size. Therefore, we resolve this instance by calling algorithm Special-WP. We thus obtain a set, $\widehat{S}$, such that $G'[(V_1 \cup V_2 \cup M) \setminus (S_1 \cup S_2 \cup \widehat{S})]$ is a $\mathcal{F}$-minor graph, and $w'(\widehat{S}) \leq \mathsf{opt}$ (since $|(V_1 \cup V_2 \cup M) \setminus (S_1 \cup S_2)| \leq n'$ and the optimal solution of each of the special subinstances is at most opt).

Observe that any obstruction in $G' - S$ is either completely contained in $G'[V_1]$, or completely contained in $G'[V_2]$, or it contains at least one vertex from M. This observation, along with the fact that $G'[(V_1 \cup V_2 \cup M) \setminus (S_1 \cup S_2 \cup \widehat{S})]$ is a $\mathcal{F}$-minor free graph, implies that $G' - T$ is a $\mathcal{F}$-minor free graph where $T = S \cup S_1 \cup S_2 \cup \widehat{S}$. Thus, it is now sufficient to show that $w'(T) \leq \frac{d}{2} \cdot (\log n')^2 \cdot \mathsf{opt}$.

By the discussion above, we have that

$$\begin{aligned} w'(T) &\leq w'(S) + w'(S_1) + w'(S_2) + w'(\widehat{S}) \\ &\leq q \cdot \log n' \cdot \mathsf{opt} + \tfrac{d}{2} \cdot ((\log n_1)^2 \cdot \mathsf{opt}_1 + (\log n_2)^2 \cdot \mathsf{opt}_2) + \mathsf{opt} \end{aligned}$$

Recall that $n_1, n_2 \leq \frac{2}{3} n'$ and $\mathsf{opt}_1 + \mathsf{opt}_2 \leq \mathsf{opt}$. Thus, we have that

$$\begin{aligned} w'(T) &< q \cdot \log n' \cdot \mathsf{opt} + \tfrac{d}{2} \cdot (\log \tfrac{2}{3} n')^2 \cdot \mathsf{opt} + \mathsf{opt} \\ &< \tfrac{d}{2} \cdot (\log n')^2 \cdot \mathsf{opt} + \log n' \cdot \mathsf{opt} \cdot (q + 1 + \tfrac{d}{2} \cdot (\log \tfrac{3}{2})^2 - \tfrac{d}{2} \cdot 2 \cdot \log \tfrac{3}{2}). \end{aligned}$$

Overall, we conclude that to ensure that $w'(T) \leq \frac{d}{2} \cdot \log^2 n' \cdot \mathsf{opt}$, it is sufficient to ensure that $q + 1 + \frac{d}{2} \cdot (\log \frac{3}{2})^2 - \frac{d}{2} \cdot 2 \cdot \log \frac{3}{2} \leq 0$, which can be done by fixing $d = \dfrac{2}{2 \log \frac{3}{2} - (\log \frac{3}{2})^2} \cdot (q + 1)$.

If we use the $\mathcal{O}(\sqrt{\log n})$-factor approximation algorithm by Feige et al. [12] for finding a balance separator in Lemma 11, then we can do the analysis similar to the deterministic case and obtain a randomized factor-$\mathcal{O}(\log^{1.5} n)$approximation algorithm for WP$\mathcal{F}$-MFD.

3 Conclusion

In this paper, we designed $\mathcal{O}(\log^{\mathcal{O}(1)} n)$-approximation algorithms for Weighted Planar $\mathcal{F}$-Minor-Free Deletion, Weighted Chordal Vertex Deletion and Weighted Distance Hereditary Vertex Deletion (or Weighted Rankwidth-1 Vertex Deletion). These algorithms are the first ones for these problems whose approximation factors are bounded by $\mathcal{O}(\log^{\mathcal{O}(1)} n)$. Along the way, we also obtained a constant-factor approximation algorithm for Weighted Multicut on chordal graphs. All our algorithms are based on the same recursive scheme. We believe that the scope of applicability of our approach is very wide. We would like to conclude our paper with the following concrete open problems.

- Does Weighted Planar $\mathcal{F}$-Minor-Free Deletion admit a constant-factor approximation algorithm? Furthermore, studying families $\mathcal{F}$ that do not necessarily contain a planar graph is another direction for further research.
- Does Weighted Chordal Vertex Deletion admit a constant-factor approximation algorithm?
- Does Weighted Rankwidth-η Vertex Deletion admit a $\mathcal{O}(\log^{\mathcal{O}(1)} n)$-factor approximation algorithm?
- On which other graph classes Weighted Multicut admits a constant-factor approximation?

References

1. Akanksha Agrawal, Daniel Lokshtanov, Pranabendu Misra, Saket Saurabh, and Meirav Zehavi. Feedback vertex set inspired kernel for chordal vertex deletion. In *Proceedings of the 28th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1383–1398, 2017.
2. Akanksha Agrawal, Daniel Lokshtanov, Pranabendu Misra, Saket Saurabh, and Meirav Zehavi. Polylogarithmic approximation algorithms for weighted-$\mathcal{F}$-deletion problems. *arXiv preprint arXiv:1707.04908*, 2017.
3. Vineet Bafna, Piotr Berman, and Toshihiro Fujito. A 2-approximation algorithm for the undirected feedback vertex set problem. *SIAM Journal on Discrete Mathematics*, 12(3):289–297, 1999.
4. Nikhil Bansal, Daniel Reichman, and Seeun William Umboh. LP-based robust algorithms for noisy minor-free and bounded treewidth graphs. In *Proceedings of the 28th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1964–1979, 2017.
5. Reuven Bar-Yehuda and Shimon Even. A linear-time approximation algorithm for the weighted vertex cover problem. *Journal of Algorithms*, 2(2):198–203, 1981.
6. Reuven Bar-Yehuda, Dan Geiger, Joseph Naor, and Ron M. Roth. Approximation algorithms for the feedback vertex set problem with applications to constraint satisfaction and Bayesian inference. *SIAM Journal on Computing*, 27(4):942–959, 1998.
7. Richard B. Borie, R. Gary Parker, and Craig A. Tovey. Automatic generation of linear-time algorithms from predicate calculus descriptions of problems on recursively constructed graph families. *Algorithmica*, 7(5&6):555–581, 1992.
8. Bruno Courcelle, Johann A. Makowsky, and Udi Rotics. Linear time solvable optimization problems on graphs of bounded clique-width. *Theory of Computing Systems*, 33(2):125–150, 2000.
9. Bruno Courcelle and Stephan Olariu. Upper bounds to the clique width of graphs. *Discrete Applied Mathematics*, 101(1-3):77–114, 2000.
10. Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012.
11. M Farber. On diameters and radii of bridged graphs. *Discrete Mathematics*, 73:249–260, 1989.
12. Uriel Feige, MohammadTaghi Hajiaghayi, and James R. Lee. Improved approximation algorithms for minimum weight vertex separators. *SIAM Journal on Computing*, 38(2):629–657, 2008.
13. Samuel Fiorini, Gwenaël Joret, and Ugo Pietropaoli. Hitting diamonds and growing cacti. In *Proceedings of the 14th Conference on Integer Programming and Combinatorial Optimization (IPCO)*, volume 6080, pages 191–204, 2010.
14. Fedor V. Fomin, Daniel Lokshtanov, Neeldhara Misra, Geevarghese Philip, and Saket Saurabh. Hitting forbidden minors: Approximation and kernelization. *SIAM Journal on Discrete Mathematics*, 30(1):383–410, 2016.
15. Fedor V. Fomin, Daniel Lokshtanov, Neeldhara Misra, and Saket Saurabh. Planar f-deletion: Approximation, kernelization and optimal fpt algorithms. In *Proceedings of IEEE 53rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 470–479, 2012.
16. Fedor V. Fomin, Daniel Lokshtanov, Venkatesh Raman, and Saket Saurabh. Bidimensionality and EPTAS. In *Proceedings of the 22nd ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 748–759, 2011.
17. Fedor V. Fomin, Daniel Lokshtanov, and Saket Saurabh. Bidimensionality and geometric graphs. In *Proceedings of the 23rd ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1563–1575, 2012.

18 Fedor V. Fomin, Daniel Lokshtanov, Saket Saurabh, and Dimitrios M. Thilikos. Bidimensionality and kernels. In *Proceedings of the 21st ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 503–510, 2010.

19 Naveen Garg, Vijay V. Vazirani, and Mihalis Yannakakis. Approximate max-flow min-(multi)cut theorems and their applications. *SIAM Journal on Computing*, 25(2):235–251, 1996.

20 Daniel Golovin, Viswanath Nagarajan, and Mohit Singh. Approximating the k-multicut problem. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 621–630, 2006.

21 Martin Charles Golumbic. *Algorithmic Graph Theory and Perfect Graphs*. Academic Press, New York, 1980.

22 Anupam Gupta, Euiwoong Lee, Jason Li, Pasin Manurangsi, and Michał Włodarczyk. Losing treewidth by separating subsets. *arXiv preprint arXiv:1804.01366*, 2018.

23 Peter L Hammer and Frédéric Maffray. Completely separable graphs. *Discrete applied mathematics*, 27(1):85–99, 1990.

24 Petr Hlinený, Sang-il Oum, Detlef Seese, and Georg Gottlob. Width parameters beyond tree-width and their applications. *The Computer Journal*, 51(3):326–362, 2008.

25 Edward Howorka. A characterization of distance-hereditary graphs. *The quarterly journal of mathematics*, 28(4):417–420, 1977.

26 Bart M. P. Jansen and Marcin Pilipczuk. Approximation and kernelization for chordal vertex deletion. In *Proceedings of the 28th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1399–1418, 2017.

27 E. J. Kim and O. Kwon. A Polynomial Kernel for Distance-Hereditary Vertex Deletion. *ArXiv e-prints*, 2016. `arXiv:1610.07229`.

28 T Leighton and S Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *Journal of the ACM*, 46:787—-832, 1999.

29 John M. Lewis and Mihalis Yannakakis. The node-deletion problem for hereditary properties is NP-complete. *Journal of Computer and System Sciences*, 20(2):219–230, 1980.

30 Carsten Lund and Mihalis Yannakakis. The approximation of maximum subgraph problems. In *Proceedings of the 20nd International Colloquium on Automata, Languages and Programming (ICALP)*, volume 700, pages 40–51, 1993.

31 John W Moon and Leo Moser. On cliques in graphs. *Israel journal of Mathematics*, 3(1):23–28, 1965.

32 G. L. Nemhauser and L. E. Trotter, Jr. Properties of vertex packing and independence system polyhedra. *Mathematical Programming*, 6:48–61, 1974.

33 Sang-il Oum. Rank-width and vertex-minors. *Journal of Combinatorial Theory, Series B*, 95(1):79–100, 2005.

34 Sang-il Oum. Approximating rank-width and clique-width quickly. *ACM Transactions on Algorithms*, 5(1), 2008.

35 Sang-il Oum. Rank-width: Algorithmic and structural results. *CoRR*, abs/1601.03800, 2016.

36 Sang-il Oum and Paul D. Seymour. Approximating clique-width and branch-width. *Journal of Combinatorial Theory, Series B*, 96(4):514–528, 2006.

37 Neil Robertson and P D Seymour. Graph minors. v. excluding a planar graph. *Journal of Combinatorial Theory Series B*, 41(1):92–114, 1986.

38 Neil Robertson and Paul D. Seymour. Graph minors .xiii. the disjoint paths problem. *Journal of Combinatorial Theory, Series B*, 63(1):65–110, 1995.

39 Neil Robertson and Paul D. Seymour. Graph minors. XX. wagner's conjecture. *Journal of Combinatorial Theory, Series B*, 92(2):325–357, 2004.

40 Horst Sachs. On the berge conjecture concerning perfect graphs. *Combinatorial Structures and their Applications*, 37:384, 1970.

41 Shuji Tsukiyama, Mikio Ide, Hiromu Ariyoshi, and Isao Shirakawa. A new algorithm for generating all the maximal independent sets. *SIAM Journal on Computing*, 6(3):505–517, 1977.

42 Mihalis Yannakakis. The effect of a connectivity requirement on the complexity of maximum subgraph problems. *Journal of the ACM*, 26(4):618–630, 1979.

43 Mihalis Yannakakis. Some open problems in approximation. In *Proceedings of 2nd Italian Conference on Algorithms and Complexity, Second (CIAC)*, pages 33–39, 1994.

A Preliminaries

For a positive integer k, we use $[k]$ as a shorthand for $\{1, 2, \ldots, k\}$. Given a function $f : A \to B$ and a subset $A' \subseteq A$, we let $f|_{A'}$ denote the function f restricted to the domain A'.

Graphs. Given a graph G, we let $V(G)$ and $E(G)$ denote its vertex-set and edge-set, respectively. In this paper, we only consider undirected graphs. We let $n = |V(G)|$ denote the number of vertices in the graph G, where G will be clear from context. The *open neighborhood*, or simply the *neighborhood*, of a vertex $v \in V(G)$ is defined as $N_G(v) = \{w \mid \{v, w\} \in E(G)\}$. The *closed neighborhood* of v is defined as $N_G[v] = N_G(v) \cup \{v\}$. The *degree* of v is defined as $d_G(v) = |N_G(v)|$. We can extend the definition of the neighborhood of a vertex to a set of vertices as follows. Given a subset $U \subseteq V(G)$, $N_G(U) = \bigcup_{u \in U} N_G(u)$ and $N_G[U] = \bigcup_{u \in U} N_G[u]$. The *induced subgraph* $G[U]$ is the graph with vertex-set U and edge-set $\{\{u, u'\} \mid u, u' \in U, \text{ and } \{u, u'\} \in E(G)\}$. Moreover, we define $G - U$ as the induced subgraph $G[V(G) \setminus U]$. We omit subscripts when the graph G is clear from context. For graphs G and H, by $G \cap H$, we denote the graph with vertex set as $V(G) \cap V(H)$ and edge set as $E(G) \cap E(H)$. An *independent set* in G is a set of vertices such that there is no edge in G between any pair of vertices in this set. The *independence number* of G, denoted by $\alpha(G)$, is defined as the cardinality of the largest independent set in G. A *clique* in G is a set of vertices such that there is an edge in G between every pair of vertices in this set.

A *path* $P = (x_1, x_2, \ldots, x_\ell)$ in G is a subgraph of G where $V(P) = \{x_1, x_2, \ldots, x_\ell\} \subseteq V(G)$ and $E(P) = \{\{x_1, x_2\}, \{x_2, x_3\}, \ldots, \{x_{\ell-1}, x_\ell\}\} \subseteq E(G)$, where $\ell \in [n]$. The vertices x_1 and x_ℓ are called the *endpoints* of the path P and the remaining vertices in $V(P)$ are called the *internal vertices* of P. We also say that P is a path between x_1 and x_ℓ or connects x_1 and x_ℓ. A *cycle* $C = (x_1, x_2, \ldots, x_\ell)$ in G is a subgraph of G where $V(C) = \{x_1, x_2, \ldots, x_\ell\} \subseteq V(G)$ and $E(C) = \{\{x_1, x_2\}, \{x_2, x_3\}, \ldots, \{x_{\ell-1}, x_\ell\}, \{x_\ell, x_1\}\} \subseteq E(G)$, i.e., it is a path with an additional edge between x_1 and x_ℓ. The graph G is *connected* if there is a path between every pair of vertices in G, otherwise G is *disconnected*. A connected graph without any cycles is a *tree*, and a collection of trees is a *forest*. A maximal connected subgraph of G is called a *connected component* of G. Given a function $f : V(G) \to \mathbb{R}$ and a subset $U \subseteq V(G)$, we denote $f(U) = \sum_{v \in U} f(v)$. Moreover, we say that a subset $U \subseteq V(G)$ is a *balanced separator for* G if for each connected component C in $G - U$, it holds that $|V(C)| \leq \frac{2}{3}|V(G)|$. We refer the reader to [10] for details on standard graph theoretic notations and terminologies that are not explicitly defined here.

Forest Decompositions. A *forest decomposition* of a graph G is a pair (F, β) where F is forest, and $\beta : V(T) \to 2^{V(G)}$ is a function that satisfies the following:

1. $\bigcup_{v \in V(F)} \beta(v) = V(G)$;
2. for any edge $\{v, u\} \in E(G)$, there is a node $w \in V(F)$ such that $v, u \in \beta(w)$;
3. for any $v \in V(G)$, the collection of nodes $T_v = \{u \in V(F) \mid v \in \beta(u)\}$ is a subtree of F.

For $v \in V(F)$, we call $\beta(v)$ the *bag* of v, and for the sake of clarity of presentation, we sometimes use v and $\beta(v)$ interchangeably. We refer to the vertices in $V(F)$ as *nodes*. A *tree decomposition* is a forest decomposition where F is a tree. For a graph G, by $\mathsf{tw}(G)$ we denote the minimum over all possible *tree decompositions* of G, the maximum size of a bag minus one in that *tree decomposition*.

Minors. Given a graph G and an edge $\{u, v\} \in E(G)$, the graph G/e denotes the graph obtained from G by contracting the edge $\{u, v\}$, that is, the vertices u, v are deleted from G and a new vertex $uv^\star$ is added to G which is adjacent to the all the neighbors of u, v previously in G (except for u, v). A graph H that is obtained by a sequence of edge contractions in G is said to be a contraction of G. A graph H is a *minor* of a G if H is the contraction of some subgraph of G. We say that a graph G is F-minor free when F is not a minor of G. Given a family $\mathcal{F}$ of graphs, we say that a graph G is $\mathcal{F}$-minor free, if for all $F \in \mathcal{F}$, F is not a minor of G. It is well known that if H is a minor of G, then $\mathsf{tw}(H) \leq \mathsf{tw}(G)$. A graph is *planar* if it is $\{K_5, K_{3,3}\}$-minor free [10]. Here, K_5 is a clique on 5 vertices and $K_{3,3}$ is a complete bipartite graph with both sides of bipartition having 3 vertices.

Chordal Graphs. Let G be a graph. For a cycle C on at least four vertices, we say that $\{u, v\} \in E(G)$ is a *chord* of C if $u, v \in V(C)$ but $\{u, v\} \notin E(C)$. A cycle C is *chordless* if it contains at least four vertices and has no chords. The graph G is a *chordal graph* if it has no chordless cycle as an induced subgraph. A *clique forest* of G is a forest decomposition of G where every bag is a maximal clique. The following lemma shows that the class of chordal graphs is exactly the class of graphs which have a clique forest.

▶ **Lemma 12** ([21]). *A graph G is a chordal graph if and only if G has a clique forest. Moreover, a clique forest of a chordal graph can be constructed in polynomial time.*

Given a subset $U \subseteq V(G)$, we say that U *intersects* a chordless cycle C in G if $U \cap V(C) \neq \emptyset$. Observe that if U *intersects* every chordless cycle of G, then $G - U$ is a chordal graph.

Improved Approximation Bounds for the Minimum Constraint Removal Problem

Sayan Bandyapadhyay[1]
Department of Computer Science, University of Iowa, Iowa City, USA
sayan-bandyapadhyay@uiowa.edu

Neeraj Kumar
Department of Computer Science, University of California, Santa Barbara, USA
neeraj@cs.ucsb.edu

Subhash Suri
Department of Computer Science, University of California, Santa Barbara, USA
suri@cs.ucsb.edu

Kasturi Varadarajan
Department of Computer Science, University of Iowa, Iowa City, USA
kasturi-varadarajan@uiowa.edu

Abstract

In the minimum constraint removal problem, we are given a set of geometric objects as obstacles in the plane, and we want to find the minimum number of obstacles that must be removed to reach a target point t from the source point s by an *obstacle-free* path. The problem is known to be intractable, and (perhaps surprisingly) no sub-linear approximations are known even for simple obstacles such as rectangles and disks. The main result of our paper is a new approximation technique that gives $O(\sqrt{n})$-approximation for rectangles, disks as well as rectilinear polygons. The technique also gives $O(\sqrt{n})$-approximation for the *minimum color path* problem in graphs. We also present some inapproximability results for the geometric constraint removal problem.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis

Keywords and phrases Minimum Constraint Removal, Minimum Color Path, Barrier Resilience, Obstacle Removal, Obstacle Free Path, Approximation

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.2

Funding Research of Sayan Bandyapadhyay, Neeraj Kumar and Subhash Suri was supported in part by the NSF grant CCF-1525817. Research of Sayan Bandyapadhyay and Kasturi Varadarajan was supported in part by the NSF grant CCF-1615845.

1 Introduction

Given a set $\mathcal{S}$ of geometric objects as *obstacles* in the plane, a path is called *obstacle-free* if it does not intersect the interior of any obstacle. In the *minimum constraint removal* (MCR) problem, the goal is to remove a minimum-sized subset $\mathcal{S}' \subseteq \mathcal{S}$ such that the remaining set $\mathcal{S} \setminus \mathcal{S}'$ admits an obstacle-free path between a source point s and the target point t. The problem is known to be NP-complete even when the obstacles have very simple geometry such as rectangles or line segments. The MCR problem is also related to the *minimum color path* (MCP) problem, where the goal is to find a path in a graph using the minimum number

[1] The work was partially done when the author was visiting University of California, Santa Barbara.

of colors. In the vertex-colored version of the problem, each vertex v of a graph $G = (V, E)$ is associated with a set of colors $\chi(v) \subseteq \mathcal{C} = \{1, 2, \ldots, |\mathcal{C}|\}$, and the goal is to find a path between two fixed vertices s and t (s-t path) that minimizes the total number of colors along the path. Similarly, in the edge-colored version, each edge of G has an associated set of colors, and the s-t path must minimize the total number of colors along the path.

The geometric constraint removal problem can be cast as a minimum color path problem by constructing a graph on the *arrangement* formed by the obstacles. The arrangement $\mathcal{A}(\mathcal{S})$ of $\mathcal{S}$ is a partition induced by the obstacles, whose *faces* are the two-dimensional connected regions, and whose edges are segments of the obstacle boundaries. We now define a planar graph $G_{\mathcal{A}}$ whose vertices are in one-to-one correspondence with the faces of the arrangement, and whose edges join two neighboring faces. By associating each obstacle with a unique color, we obtain a version of the minimum color path problem—an s–t path has exactly as many colors along it as the number of obstacles it crosses. However, it is worth pointing out that the number of vertices in $G_{\mathcal{A}}$ can be *quadratic* in the size of the geometric input: a set of n geometric obstacles, each with a constant number of boundary edges, can create an $\Omega(n^2)$ size arrangement.

The minimum color path problem is also NP-complete, and by a reduction from Set Cover it is also NP-hard to approximate to a factor better than $o(\log n)$ even if the graph is planar [8, 14, 20]. In [13], Hassin et al. solve a special case of the MCP problem where each edge has *exactly* one color. For that special case, they present an $O(\sqrt{|V|})$-approximation algorithm. However, for the general MCP problem, no sub-linear algorithm appears to be known to the best of our knowledge.

The geometric minimum constraint removal problem has been studied under different names across multiple research communities, including sensor networks and robotics. In the sensor networks, the problem is called *barrier resilience*, in which a collection of sensors are modeled for providing (overlapping) geometric coverage in the plane, and the network's resilience is measured by the minimum number of sensors whose *removal* creates a *sensor-avoiding* s–t path. The most common form of geometric obstacles considered in sensor network applications are *circular disks*. When all disks have the same (unit) radius, a 2-approximation is known due to Chan and Kirkpatrick [5], who build and improve upon the earlier work of Bereg and Kirkpatrick [3]. However, even for this simple case, the complexity of minimum constraint removal is an unsolved open problem [5]. In [5, 17], constant factor approximations are proposed for *restricted versions* of arbitrary radii disks. However, in general, when disks have arbitrary radii, no sub-linear approximation with provable guarantee is known. The problem has also been studied for other types of obstacles, mainly from the perspective of time complexity. The problem has been shown to be NP-complete for convex obstacles [11], for line segments [19], even in the bounded density case [2, 10], and for axis-parallel rectangles with aspect ratio close to one [17].

In robotics, the minimum constraint removal problem models the motion planning problem of multi-articulated robot [11, 14]. Suppose we have a physical environment constaining a disjoint set of impenetrable obstacles in the plane, and a robot with two degrees of freedom. Then the *configuration space* approach to motion planning *shrinks* the robot to a point while simultaneously *expanding* the obstacle by taking their Minkowski sum with the robot's geometry. The result is our minimum constraint removal problem: a set of two-dimensional intersecting obstacles that may have no feasible path for the robot, and so some obstacles need to be removed.

Finally, the problem has also been studied through the lenses of parameterized complexity [10, 17], and exact and heuristic algorithms [9, 14]. It is also loosely related to a shortest path problem in the plane [1, 15], where given a set of disjoint obstacles, the goal is to find an Euclidean shortest s-t path that intersects at most k obstacles.

1.1 Our Results

In this work, we make progress on both the graph and the geometric versions of MCP by obtaining improved approximation results. All these approximations are achieved in polynomial time. For the minimum constraint removal problem, we obtain the following results.

- We present an $O(\sqrt{n})$-approximation for rectilinear polygons, where n denotes the total number of vertices of the polygons. No sublinear approximation was known even for squares.
- We present an $O(\sqrt{n})$-approximation for disks, where n denotes the number of disks. For arbitrary disks, the only approximation results known are in the restricted cases, where either the crossing patterns of the paths are limited or the aspect ratio and the density are bounded.

Prior approximation algorithms for MCR proceed by establishing a good bound on the number of times an optimal path enters a removed obstacle. For the obstacles we consider, i.e., rectilinear polygons and disks, this bound is very large. Thus the previous approaches are inadequate to obtain the above mentioned results. Our main new idea is to use a filtering step, which removes a small number of obstacles that are potentially expensive in terms of the number of times an optimal path enters those obstacles.

The above results are based on an algorithmic framework, which uses the filtering step mentioned before. As a byproduct, the framework also gives an $O(\sqrt{|V|})$-approximation for MCP on vertex-colored graphs.

We also obtain a few hardness results for MCR which give a better understanding of the problem. We show that for rectilinear polygons, the problem is NP-hard to approximate within a factor better than 2. The same result holds even for convex polygons. We also prove the APX-hardness of the problem in a more restricted case, where the obstacles are axis-parallel rectangles.

The framework is described in Section 2. The application of the framework to the MCR problem is discussed in Section 3. Finally we describe the hardness results in Section 4. Throughout this paper, the proofs of lemmas and theorems marked with $(*)$ are given in the appendix due to space constraints.

2 An Algorithmic Framework

We begin our discussion by introducing a generic framework that yields a sublinear approximation for minimum color path problems on graphs. In the later sections, we apply this framework to achieve similar approximation bounds for the MCR problem. Roughly speaking, the framework comprises of two main steps. In the first step, a 'small' subset of the colors are removed from the instance based on some conditions. In the second stage, an approximation of the minimum color path is computed using a shortest path algorithm. We start with some basic definitions.

As an input to the framework, we assume that we are given a graph $G = (V, E)$, the source vertex s, the target vertex t, and a set of colors $\mathcal{C}$, such that each vertex $v \in V$ is assigned a subset $\chi(v) \subseteq \mathcal{C}$ of colors. We will refer to such a graph as a *colored graph* and denote it by $G = (V, E, \mathcal{C})$. we define the set of colors $\chi(\pi)$ used by π to be the union of the colors of vertices on this path. That is, $\chi(\pi) = \bigcup_{v \in \pi} \chi(v)$.

▶ **Definition 1.** Any path π in G is a *k-color path* if the number of colors used $|\chi(\pi)|$ is k.

An algorithm is called an *α-approximation* algorithm for computing a k-color path if it satisfies the following two conditions: (1) if there exists a k-color path π from s to t, it computes a path π^* such that $|\chi(\pi^*)| \leq \alpha k$, and (2) if there is no k-color path from s to t, it returns an arbitrary s-t path. The following is straightforward.

▶ **Lemma 2.** *If there exists an α-approximation algorithm to compute a k-color path from s to t then there also exists an α-approximation algorithm for computing a minimum color path from s to t.*

Proof. We try all possible values $k = 1, 2, \ldots, |\mathcal{C}|$ and let π_k be the path returned by the approximation algorithm for computing a k-color path for a given value of k. Let j be the value such that $\chi(\pi_j)$ has smallest cardinality over all $\chi(\pi_k)$. Now, let l be the number of colors used by a minimum color path, then $|\chi(\pi_l)|$ must be at most αl. Clearly, $|\chi(\pi_j)| \leq |\chi(\pi_l)| \leq \alpha l$ and therefore π_j is an α-approximation for computing a minimum color path. ◀

From Lemma 2 it follows that computing an approximation of a k-color path is sufficient, and therefore in the rest of our discussion, we work towards that goal. Next, we describe the details of our framework.

2.1 Approximation Framework

As an input to the framework, we assume that we are given a colored graph $G = (V, E, \mathcal{C})$, and an integer k. The key idea behind our approximation framework is to define a notion of *neighborhood* for the colors in $\mathcal{C}$, and 'discard' the colors that have dense neighborhoods.

▶ **Definition 3.** Let $\mathcal{P}$ be an arbitrary set of objects and β be a parameter. We define *neighborhood* $\mathcal{N} : \mathcal{C} \to 2^{\mathcal{P}}$ to be a mapping from $\mathcal{C}$ to subsets of $\mathcal{P}$ that satisfies the following properties.

1. ***(Bounded-Size Property)*** Sum of cardinalities of all neighborhoods $\sum_{C \in \mathcal{C}} |\mathcal{N}(C)|$ is $O(k\beta^2)$
2. ***(Bounded-Occurrence Property)*** If there exists a k-color path in G, then there also exists a k-color path π^* in G such that, for any color $C \in \mathcal{C}$, the number of times C appears on π^* is at most $O(|\mathcal{N}(C)|)$.

The set $\mathcal{P}$ in the above definition can be any set of objects. For example, in MCP problem $\mathcal{P}$ is the set of vertices of the graph. In the geometric MCR, $\mathcal{P}$ is a set of points in the plane. We now describe our approximation algorithm which we will refer to as APPROX-CORE.

Algorithm APPROX-CORE

1. Construct the neighborhood $\mathcal{N}(C)$ for each color $C \in \mathcal{C}$.
2. For all $C \in \mathcal{C}$, remove all occurrences of the color C from the graph G if $|\mathcal{N}(C)| \geq \beta$. Let G' be the modified graph after removing all such colors.
3. For every vertex v in G', assign an integer weight $|\chi(v)|$ on v.
4. Compute a minimum weight path π from s to t in G' using Dijkstra's Algorithm. Return π.

▶ **Lemma 4.** *Given the set $\mathcal{P}$ and a parameter β, the algorithm APPROX-CORE gives an $O(\beta)$-approximation for the k-color path in G.*

Proof. Assume that there exists a k-color path in G. Otherwise, the proof is trivial as the algorithm always returns a path. Let $\mathcal{C}_1$ be the set of colors removed during step 2 of the algorithm, and $\mathcal{C}_2$ be the set of colors in G' that appear on the path π returned by the algorithm. Then, the total number of colors in G that may appear on π is at most $|\mathcal{C}_1| + |\mathcal{C}_2|$.

First we compute a bound on the size of $\mathcal{C}_1$. Observe that the neighborhood of each color $C \in \mathcal{C}_1$ has size at least β. Therefore, we have:

$$\begin{aligned} \sum_{C \in \mathcal{C}_1} |\mathcal{N}(C)| &\leq \sum_{C \in \mathcal{C}} |\mathcal{N}(C)| \\ \implies |\mathcal{C}_1| \cdot \beta &\leq O(k\beta^2) && \text{By the bounded-size property of } \mathcal{N} \\ \implies |\mathcal{C}_1| &\leq ck\beta && \text{for some constant } c \end{aligned}$$

Next, we compute a bound on size of $\mathcal{C}_2$. Towards this end, observe that the neighborhood $\mathcal{N}(C)$ of every color C in G' has fewer than β colors. By the bounded-occurrence property of the neighborhood $\mathcal{N}$, there exists a k-color path π^* in G such that for each $C \in \mathcal{C}$, the number of times C appears on π^* is at most $O(|\mathcal{N}(C)|)$. Therefore, it follows that any color C in G' appears on π^* at most $O(\beta)$ times. In other words, there exists a path in G' that has weight at most $c'k\beta$ for another constant c'. Therefore the number of colors used by the minimum weight path π is at most $c'k\beta$.

Hence, the total number of colors in $\mathcal{C}$ that appear on π is at most $|\mathcal{C}_1| + |\mathcal{C}_2| = (c + c') \cdot k\beta$, which is an $O(\beta)$-approximation. ◀

We obtain the following theorem.

▶ **Theorem 5.** *Given a colored graph $G = (V, E, \mathcal{C})$, suppose a neighborhood $\mathcal{N}$ for G can be constructed in polynomial time that satisfies the bounded-size and bounded-occurrence property. Then there exists a polynomial time algorithm that achieves an $O(\beta)$-approximation for computing a k-color path in G.*

Therefore, in order to achieve an approximation for the k-color path, it just suffices to construct a neighborhood $\mathcal{N}$, that satisfies the bounded-size and bounded-occurrence properties. In the next section, we illustrate this construction for MCP on vertex-colored graphs.

2.2 Application to Minimum Color Path

In this section, we will apply the above framework to achieve $O(\sqrt{n})-$approximation for MCP on a vertex-colored graph $G = (V, E, \mathcal{C})$ with n vertices. Our goal is to simply compute a neighborhood $\mathcal{N}$ for a k-color path in G such that $\mathcal{N}$ has bounded-size $O(kn)$ and satisfies the bounded-occurrence property. Using Lemma 2 and $\beta = \sqrt{n}$ in Theorem 5, an $O(\sqrt{n})$-approximation for MCP follows.

We define neighborhood $\mathcal{N}(C)$ of each color C to be the set $\{v \in V \mid C \in \chi(v) \text{ and } |\chi(v)| \leq k\}$. The bounded-occurrence property is easily satisfied because a k-color path π_k will never visit vertices that contain more than k colors, and since π_k is simple, each occurrence of a color C on the path can be uniquely charged to a vertex in $\mathcal{N}(C)$. To see that the bounded-size property is satisfied, we note the following.

$$\sum_{C \in \mathcal{C}} |\mathcal{N}(C)| = \sum_{v \in V : |\chi(v)| \leq k} |\chi(v)| \leq O(kn).$$

▶ **Theorem 6.** *There exists an $O(\sqrt{n})$-approximation algorithm for MCP on vertex-colored graphs.*

Application to Minimum Label Path. As another example application for the framework, we consider a special case of MCP when each edge has exactly one color (called its label). This problem has been well studied [4,12,13] under the name *minimum label path*. Hassin et al. [13] gave an $O(\sqrt{n})$-approximation for this problem on general graphs. Using our framework and the following simple definition of neighborhood, we can achieve an $O(\sqrt{\frac{n}{OPT}})$-approximation if the number of edges in G is $O(n)$. Here OPT is the number of labels used by any minimum label s-t path.

For the sake of applying the framework, we transform the input edge-colored graph $G = (V, E, \mathcal{C})$ into a vertex-colored graph H by adding a vertex corresponding to each edge. The color corresponding to an old edge is moved to the new vertex. Now, for each new vertex v that has color C, we include both neighbors (old vertices) of v in H to the neighborhood of C. The bounded-occurrence property is straightforward. For the bound on size, observe that an old vertex v can be in at most $degree(v)$ neighborhoods, so sum of cardinality of all neighborhoods is at most $2|E|$. Since $|E| = O(n)$, the size of $\mathcal{N}$ is $O(n) = O(\frac{n}{k} \cdot k)$. With $\beta = \sqrt{n/k}$, Theorem 5 and Lemma 2 give an $O(\sqrt{\frac{n}{OPT}})$-approximation.

3 Application to Geometric Objects

In this section, we apply our approximation framework to achieve sublinear approximation for MCR when the obstacles are rectilinear polygons (Section 3.1) and disks of arbitrary radii (Section 3.2). Observe that if m is the number of cells of the input arrangement $\mathcal{A}$ of obstacles in $\mathcal{S}$, applying Theorem 6 on the graph obtained from $\mathcal{A}$ easily gives a $O(\sqrt{m})$-approximation. However, m can be $\Omega(n^2)$ and therefore this approach does not give us an $O(\sqrt{n})$-approximation. Here n is the number of vertices if obstacles are polygonal or the number of disks otherwise. By exploiting the geometry of obstacles, we show how to construct a colored graph $G = (V, E, \mathcal{C})$ and a sparse neighborhood $\mathcal{N}$ even when the underlying colored graph G can have $\Omega(n^2)$ complexity.

Recall that, there are two main steps for applying the framework. First we need to construct the colored graph G such that an s–t path in the plane that removes the minimum number of constraints, corresponds to a path in G that uses the minimum number of colors. Next, we need to construct the neighborhood $\mathcal{N}$ for colors in G such that it satisfies the bounded-size and bounded-occurrence properties. Note that for technical reasons, the graph G we construct for the geometric instances has colors assigned on edges. Indeed, for the sake of applying the framework, one can easily transform it into a vertex-colored graph by adding a vertex corresponding to each edge. We begin by revisiting some necessary background.

Any arrangement of obstacles in the plane can be partitioned into two distinct regions namely the obstacles, and *free space*, that is the region of the plane not occupied by obstacles. Without loss of generality, we assume that the points s and t lie in free space, as we must remove all the obstacles that are incident to either s or t in order to find an obstacle free s–t path. We say that a path π *crosses* an obstacle S if π intersects the interior of S. Note that, as s and t lie in free space, if π crosses S, π must intersect the boundary of S transversally.

Consider an optimal path π that removes the minimum number of obstacles. It is easy to see that π will cross an obstacle S if and only if S was removed from input. Therefore, removing an obstacle is equivalent to crossing it. In the following, we introduce the notion of a *k-crossing path*.

▶ **Definition 7.** A path π in the plane is called a *k-crossing path* if it crosses exactly k obstacles.

It is easy to see that if each obstacle is assigned a unique color and we assign color to a path whenever it enters an obstacle, then a k-crossing path π uses exactly k colors. Observe that although the space of k-crossing paths is infinite, we want to establish a one to one correspondence between the path in the plane that crosses minimum number of obstacles and a path in G that uses the minimum number of colors.

▶ **Definition 8.** Given a set $\mathcal{S}$ of obstacles in the plane, a one to one mapping $\mathcal{M} : \mathcal{S} \to \mathcal{C}$ from a set of obstacles $\mathcal{S}$ to a set of colors $\mathcal{C} = \{1, 2, \ldots, |\mathcal{C}|\}$, we say that a graph $G = (V,\ E,\ \mathcal{M}(\mathcal{S}))$ with two fixed vertices v_s, v_t is a *valid colored graph* for the input arrangement if the following conditions hold:

1. if there is a k-color v_s-v_t path in G, then there is also a j-crossing s-t path in the plane for some $j \leq k$, and
2. if there is a k-crossing s-t path in the plane, then there is also a j-color path from v_s to v_t in G for some $j \leq k$.

The first condition is typically established by fixing an *embedding* for the edges of G in the plane. From the above discussion and using Lemma 2 and Theorem 5 with $\beta = \sqrt{n}$, we have the following.

▶ **Lemma 9.** *Suppose we are given a valid colored graph $G = (V, E, \mathcal{C})$ for an arrangement of the set $\mathcal{S}$ of input obstacles in the plane, such that there is a k-color path in G. If we can construct the neighborhoods $\mathcal{N}(S)$ for all obstacles $S \in \mathcal{S}$ such that, the total size across all neighborhoods is $O(kn)$* (bounded-size)*, and there exists a k-color path π in G from v_s to v_t such that for any obstacle $S \in \mathcal{S}$, the corresponding color appears on π at most $O(|\mathcal{N}(S)|)$ times* (bounded-occurrence)*, then* APPROX-CORE *achieves an $O(\sqrt{n})$-approximation for* MCR*.*

3.1 An $O(\sqrt{n})$-approximation for Rectilinear Polygons

We begin by describing our construction of a valid colored graph $G = (V,\ E,\ \mathcal{M}(\mathcal{S}))$ for the input set of obstacles $\mathcal{S}$. Without loss of generality, we assume that the mapping $\mathcal{M}$ simply assigns a unique color $C_i \in \mathcal{C}$ to each obstacle $S_i \in \mathcal{S}$.

Graph Construction. Let V be the set of vertices of all obstacles in $\mathcal{S}$ (including s and t). Let v_s and v_t be the vertices corresponding to the points s and t, respectively. We build a complete graph over this vertex set by adding an edge (u, v) to E for every pair of vertices $u, v \in V$. We define a *rectilinear embedding* of an edge $e = (u, v)$ in the plane as follows.

> Without loss of generality, assume u lies below and to the left of v, and let x be the point where a horizontal ray from u and a vertical ray from v intersect. The rectilinear path $\pi_{uv} = \overline{ux} \to \overline{xv}$ is called the embedding of edge e. We assign e the colors corresponding to obstacles, whose boundaries are intersected by π_{uv} transversally.

Roughly speaking, we assign a color to an edge if it intersects both the interior and boundary of the corresponding obstacle, i.e., when the edge enters or exits the obstacle. It is easy to see that with the above construction, the first condition of Definition 8 is satisfied. For the second requirement, we make the following claim.

▶ **Lemma 10.** *If there exists a k-crossing s-t path π^* in the plane, then there exists a v_s-v_t path π in G that uses at most k colors.*

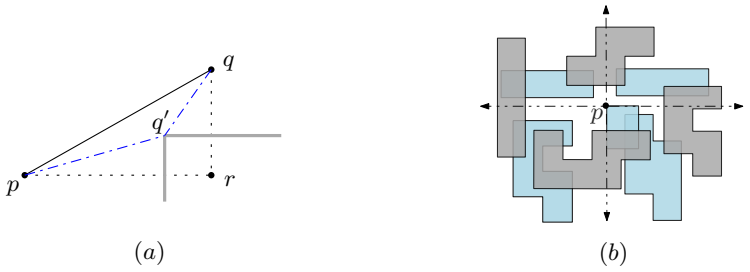


Figure 1 (a) Recursively modifying the path π to make it rectilinear. Observe that the segment $\overline{pq}$ cannot cross any obstacle edge. (b) Construction of the neighborhood for $k = 1$. p lies in the neighborhoods of all polygons shown in dark gray.

Proof. We prove this in three steps. First we construct a path π' from π^* such that π' consists of only straight line segments. Next we modify π' to obtain a rectilinear path $\pi_\perp$. Then we claim that $\pi_\perp$ is corresponding to a path π in G that uses at most k colors.

First, we remove all the k obstacles that are crossed by π^* which makes π^* lie entirely in free space connecting s and t. Since s and t are now connected in free space, there must exist a path of minimum Euclidean length that also lies in free space. Let this path be π'. It is easy to see that edges of π' are straight line segments, and lie in free space. Also π' bends only at obstacle vertices. We now recursively modify each segment $\overline{pq}$ of π to obtain a rectilinear path $\pi_\perp$. Let r be the point where a horizontal line through p intersects a vertical line through q. There are two cases. (See also Figure 1(a).)

1. If the triangle Δpqr contains one or more obstacle vertices, we find the vertex q' closest to the segment $\overline{pq}$, and replace $\overline{pq}$ with the segments pq' and $q'q$. It is easy to verify that π still lies in free space after this modification. We now repeat the process recursively on the segments pq' and $q'q$.
2. Otherwise, we simply replace $\overline{pq}$ by the rectilinear path $\pi_{pq} = pr \to rq$. Observe that since the obstacles are rectilinear and the segment $\overline{pq}$ lies in free space, no obstacle segment can intersect the triangle Δpqr, and therefore π_{pq} also lies in free space.

Observe that $\pi_\perp$ obtained using the above procedure lies in free space and crosses the boundaries of no more than k obstacles. Next, we retrieve an v_s-v_t path in G from this rectilinear path $\pi_\perp$. Note that, the way $\pi_\perp$ is constructed, at least one of the two endpoints of each of its segments is a vertex of a polygon and hence appears as a vertex in G. It follows that, the subpath between two such consecutive vertices along $\pi_\perp$ must consists of at most two rectilinear segments: one horizontal segment followed by a vertical segment. Hence, this subpath is the rectilinear embedding of the edge between the two vertices. We construct a path π by selecting the vertices along $\pi_\perp$ in order and connecting each consecutive pair of vertices by an edge. By definition, π is in G and uses no more than k colors. ◀

Construction of the Neighborhood. Now we construct the neighborhood $\mathcal{N}(S)$ for all obstacles $S \in \mathcal{S}$ that satisfies the bounded-size and bounded-occurrence properties. We choose the ground set $\mathcal{P}$ of elements in the neighborhood to be the set of vertices V of the obstacles in $\mathcal{S}$. Now, we define the *neighborhood* $\mathcal{N}(S)$ of an obstacle $S \in \mathcal{S}$ to be the subset of vertices from where one can reach a point on the boundary of S moving along a vertical or a horizontal segment and crossing no more than k obstacles. Roughly speaking, $\mathcal{N}(S)$ comprises of the vertices 'nearby' the boundary of S. We compute the set $\mathcal{N}(S)$ for every $S \in \mathcal{S}$ as follows.

For every vertex $v \in V$, draw four axis-aligned rays emanating at v one in each of the four directions. Next, find the first k distinct obstacles whose boundaries are intersected by each of these rays transversally as we move away from v. Let this set be X_v and therefore $|X_v| \leq 4k$. For each $S \in X_v$, include v to the neighborhood $\mathcal{N}(S)$. (See also Figure 1.(b))

The elements in the union of all neighborhoods is the set of vertices in the input and therefore has size $O(n)$. Moreover, since each element is present in at most $4k$ neighborhoods, $\sum_{S\in\mathcal{S}} |\mathcal{N}(S)|$ is $O(kn)$. Therefore the *bounded-size* property is easily satisfied. For the bounded-occurrence property, we prove the following lemma.

▶ **Lemma 11.** *Let π be any k-color v_s-v_t path in G and $S_i \in \mathcal{S}$ be an arbitrary obstacle. Then the color C_i corresponding to obstacle S_i appears on edges of π at most $O(|\mathcal{N}(S_i)|)$ times.*

Proof. We consider the set $E_i = \{e \mid C_i \in \chi(e)\}$ of edges that contain the color C_i and need to show that $|E_i| = O(|\mathcal{N}(S_i)|)$. Let $e = (p, q)$ be an arbitrary edge in E_i and let $\pi_{pq} = \overline{pr} \rightarrow \overline{rq}$ be its rectilinear embedding. From our construction, e has color C_i iff π_{pq} crossed S_i. Therefore, at least one of $\overline{pr}$ and $\overline{rq}$ must intersect the boundary of S_i transversally. This implies that at least one of p and q must be included in $\mathcal{N}(S_i)$ during our neighborhood construction. If $p \in \mathcal{N}(S_i)$, we charge this occurrence of color C_i to p, otherwise we charge it to q. Since a vertex $p \in \mathcal{N}(S_i)$ is adjacent to at most two edges in π, every element in $\mathcal{N}(S_i)$ is charged at most twice. Therefore C_i occurs on edges of π at most $2|\mathcal{N}(S_i)|$ times. ◀

Using Lemma 9, we obtain the following result.

▶ **Theorem 12.** *If all the obstacles in $\mathcal{S}$ are rectilinear polygons, then there exists an $O(\sqrt{n})$-approximation algorithm for the MCR problem.*

3.2 An $O(\sqrt{n})$-approximation for Arbitrary Disks

We will now consider the case when all the input obstacles are disks, of possibly different radii. The construction of the neighborhood for disks needs to be different, as the earlier arguments for rectilinear polygons relied heavily on obstacles having corners and therefore do not apply to disks. Recall that in order to apply our approximation framework, we first need to construct a valid colored graph $G = (V, E, \mathcal{C})$ (Lemma 9). Towards this end, we simply let G to be the graph $G_{\mathcal{A}}$ induced by the input arrangement $\mathcal{A}$: each cell of $G_{\mathcal{A}}$ contains a vertex and any pair of neighboring cells (vertices) are joined by an arc that does not intersect any other cell. Let v_s and v_t be the vertices in G corresponding to the cells that contain s and t, respectively. We assign a unique color to each disk. Additionally, we make G directed by replacing each edge by two directed edges. For each directed edge $e = (u, v)$, we assign to e the set of colors corresponding to all the disks D such that v lies in the interior of D and u does not lie in the interior of D. Roughly speaking, we assign colors when the edge *enters* into an obstacle.

Note that the way G is defined, it is a plane graph and we consider its natural embedding which is also planar. Since we assign colors when an edge of G enters an obstacle, it is easy to see that a k-color path π in G corresponds to a k-crossing path π' in the plane. For the other direction, given a k-crossing path π' we can easily construct a path π in G by simply concatenating the vertices corresponding to each arrangement cell intersected by π' in order. Thus, we have the following immediate observation.

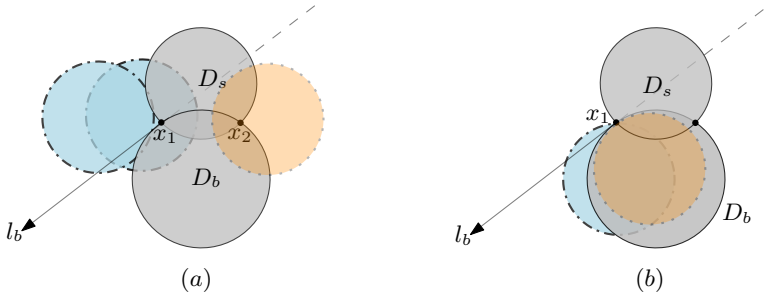


Figure 2 Disks shown in dash-dotted (shaded blue) are *critical* and included to $\mathcal{D}(D_s, D_b)$. Disks shown in dotted (shaded orange) are not critical with respect to the pair D_s, D_b.

▶ **Observation 13.** *G is a valid colored graph for the input disks.*

As each color is corresponding to a disk and vice versa, we will use the term disk instead of color in the context of applying the framework. In the following, we describe the neighborhood construction. The key idea behind our construction is to pick a subset of the $O(n^2)$ possible intersection points between pairs of disks in the input and use them for the ground set $\mathcal{P}$ of the neighborhood.

Computation of the Set $\mathcal{P}$ and the Neighborhood $\mathcal{N}$

Consider any two disks D_s, D_b such that their boundaries intersect transversally. Without loss of generality, assume that radius of D_s is smaller than D_b and their boundaries intersect at points x_1, x_2. Consider the arc $\overset{\frown}{x_1x_2}$ of ∂D_b (boundary of D_b) that lies inside D_s and assume x_1 lies before x_2 in clockwise traversal of this arc. Let l_b be the half line emanating at x_1 colinear to the tangent of D_b at x_1 and not intersecting D_s as shown in Figure 2. We now define the set of *critical* disks $\mathcal{D}(D_s, D_b)$ corresponding to a pair of intersecting disks D_s and D_b.

▶ **Definition 14** (Critical Disks)**.** Let D be a disk that intersects both D_s and D_b. We say that the disk D is *critical* with respect to the pair D_s, D_b if D intersects the half line l_b (see Figure 2).

We include the tuple (x_1, D_s, D_b) to the set of neighborhood candidates $\mathcal{P}$ if and only if the size of the critical set $\mathcal{D}(D_s, D_b)$ corresponding to the intersecting pair D_s, D_b is at most k. We are now ready to define the neighborhood $\mathcal{N}(D)$ of a disk D.

> For each disk D, add all the tuples of the form (x_1, D_s, D) to the neighborhood $\mathcal{N}(D)$. In addition, add a constant number of *phantom* points to the neighborhood $\mathcal{N}(D)$ of all disks D.

In the next few lemmas we establish the bounded-size and bounded-occurrence property of $\mathcal{N}$. We will need the notion of *exterior-disjointness*. Given a disk D and two disks D_1, D_2 that intersect D, we say that D_1 and D_2 are exterior-disjoint w.r.t D if the regions $D_1 \setminus D$ and $D_2 \setminus D$ are disjoint. A set $\mathcal{D}$ of disks is called exterior-disjoint w.r.t a disk D if any pair of disks in $\mathcal{D}$ is exterior-disjoint of D. We note the following.

▶ **Lemma 15.** *For a given disk D_s, let $\mathcal{D}$ be a set of disks such that the radius of any disk in $\mathcal{D}$ is at least the radius of D_s. If $\mathcal{D}$ is exterior-disjoint w.r.t D_s, then $|\mathcal{D}|$ is at most six.*

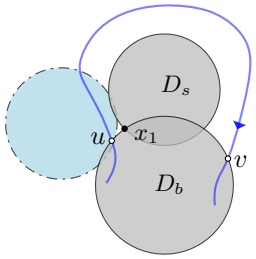


Figure 3 An entry of π_{uv} is charged to $x_1 \in \mathcal{N}(D_b)$.

Now we prove the bounded-size property of $\mathcal{N}$. We note that, if there are no three disks in the input whose boundaries intersect at a common point, then this proof follows readily from the bound of $O(kn)$ on the number of depth k points in an arrangement of disks [7, 16]. To see this, observe that for a tuple (x_1, D_s, D_b), that is added to $\mathcal{P}$, the intersection point of D_s and D_b must lie inside at most k other disks. Otherwise, the set $\mathcal{D}(D_s, D_b)$ contains more than k disks which is a contradiction. Moreover, since no more than two disks intersect at a point, each such intersection point adds at most two tuples. Thus the total size of all neighborhoods is at most two times the number of vertices in the arrangement of the input disks that lie inside at most k other input disks. From [7] and [16], it follows that the number of such vertices is $O(kn)$, and hence the bounded-size property follows. However, in presence of degeneracy, we might add as many as k tuples corresponding to one intersection point, and thus the bound does not follow immediately. Nevertheless, we prove the following lemma.

▶ **Lemma 16.** $(*)$ *The number of elements in the the set $\mathcal{P}$ is $O(kn)$.*

Since each element of $\mathcal{P}$ is included in a unique neighborhood, the bounded-size property of $\mathcal{N}$ follows from the above lemma. Next, we prove the bounded-occurrence property of $\mathcal{N}$. Observe that, it is sufficient to show the existence of a k-crossing s-t path π in the plane that enters into (and therefore crosses) any disk D at most $|\mathcal{N}(D)|$ times. This is because, an edge of G is assigned the color C corresponding to D only when it enters D, and thus every occurrence of C on the path π is corresponding to a crossing of D by π. Also existence of such a geometric k-crossing s-t path π implies the existence of a k-color v_s-v_t path in G where each color corresponding to a disk D appears at most $|\mathcal{N}(D)|$ times.

Now consider any k-crossing path π in the plane. By a similar argument as in Lemma 10, there also exists a k-crossing path π' such that the edges of this path are either straight line segments (tangents between a pair of disks) or parts of obstacle boundary (arcs). With this convention, one can now easily define the length of any such path as the sum of the lengths of its segments and arcs. Let π^* be the minimum length s-t path that crosses k disks. Then, we prove the following lemma.

▶ **Lemma 17.** *π^* crosses any disk D_b at most $O(|\mathcal{N}(D_b)|)$ times.*

Proofsketch. It suffices to prove that every entry (and therefore crossing) of the disk D_b by π^* can be charged to an element in $\mathcal{N}(D_b)$ so that every element is charged at most $O(1)$ times. Let $v_1, v_2, \ldots, v_l$ be the entry points on the disk D_b by π^*. We charge the first entry to a phantom point. Now consider the i^{th} crossing for any $i > 1$. Let u_i be the point on π^* immediately before v_i where π^* last crossed D_b. Thus the subpath $\pi_{u_i v_i}$ of π^* between u_i and v_i lies in the exterior of D_b. Let $A_{u_i v_i}$ be the arc on the boundary of D_b with endpoints u_i and v_i such that the region $R_{u_i v_i}$ enclosed by the closed curve $\pi_{u_i v_i} \cup A_{u_i v_i}$ does not contain D_b.

Consider the set of arcs $\{A_{u_iv_i} \mid 2 \leq i \leq l\}$. First, assume that any two arcs in this set are disjoint. Since π^* is a minimum length k-crossing path, there must be at least one disk not crossed by π^* that intersects $R_{u_iv_i}$ as well as the disk D_b. Let D_s be the first such disk encountered while traversing the arc $A_{u_iv_i}$ clockwise along the boundary of D_b. There are two cases. If D_s is bigger than D_b, we charge the crossing to one of the phantom points in $\mathcal{N}(D_b)$. Otherwise, we assume D_s is smaller than D_b. Let x_1 be the first intersection point of ∂D_s and ∂D_b encountered while traversing the arc $A_{u_iv_i}$ clockwise along the boundary of D_b. Observe that π^* must cross all the disks in $\mathcal{D}(D_s, D_b)$ because otherwise it will contradict the choice of D_s.

Hence the size of $\mathcal{D}(D_s, D_b)$ is bounded by k, the number of disks that π^* can cross. This implies the tuple (x_1, D_s, D_b) must be included to the neighborhood $\mathcal{N}(D_b)$, and therefore we can charge this crossing to this tuple.

Because of the disjointness assumption of the arcs, the same tuple cannot be charged to another crossing.

Also it is easy to verify that the phantom points need not be charged more than a constant number of times, as the set of disks bigger than D_b for which we charge a phantom point must be exterior-disjoint of D_s and therefore can be at most six (Lemma 15). The case when the arcs are not disjoint, for any two arcs, one must contain the other. This is true, as the subpaths $\pi_{u_iv_i}$ as defined above cannot intersect each other. By exploiting this structure of the arcs one can prove the lemma in this case as well. ◀

Using Lemma 9, we obtain the following result.

▶ **Theorem 18.** *If all the obstacles in $\mathcal{S}$ are disks, then there exists an $O(\sqrt{n})$-approximation algorithm for the MCR problem.*

Using a different realization of the algorithmic approach described above, it appears possible to derive an approximation guarantee close to $O(\sqrt{n})$ for other obstacle types. We sketch this for the case where the obstacle set $\mathcal{S}$ is a set of triangles satisfying standard degeneracy assumptions. We define the *level* of a 2-dimensional cell σ in the arrangement of the triangles in $\mathcal{S}$ to be the minimum number of triangles whose removal results in an obstacle-free path from s to (any point in) σ. Thus, there is only one cell at level 0, and this is the cell containing s. We can show that the number of cells with level at most k is $O(kn\alpha(n))$, where $\alpha(\cdot)$ is the inverse Ackermann function. Furthermore, the number of arrangement edges bordering such cells is also $O(kn\alpha(n))$.

Suppose there is a k-crossing path from s to t, and we want to approximate it. For a triangle $T \in \mathcal{S}$, we include in its neighborhood $\mathcal{N}(T)$ any arrangement edge e such that (a) e is part of T's boundary, and (b) e borders a cell σ that is contained in T and has level at most k. It follows that the sum of the neighborhood sizes is $O(kn\alpha(n))$. We can also establish the bounded occurrence property, leading to an approximation guarantee of $O(\sqrt{n\alpha(n)})$. We defer the details to the journal version.

4 Hardness of Approximation

In this section, we describe the 2-inapproximability and the APX-hardness results for rectilinear polygons and axis-parallel rectangles, respectively. Due to space constraints, we will just mention the results and defer the details to the appendix.

2-inapproximability. We reduce an instance of Vertex Cover to an instance of MCR with rectilinear polygons. Since Vertex Cover is hard to approximate within a factor of 2 assuming the Unique Games conjecture [18], we get the following theorem.

▶ **Theorem 19.** *Minimum constraint removal with rectilinear polygons is hard to approximate within a factor of* 2 *assuming the Unique Games conjecture.*

The same construction can also be extended for convex polygons.

▶ **Corollary 20.** *Minimum constraint removal with convex polygons is hard to approximate within a factor of* 2 *assuming the Unique Games conjecture.*

APX-hardness for Axis Parallel Rectangles. We reduce a restricted version of vertex cover to our problem which is referred to as Special-3VC. Chan et al. [6] had introduced this version for the sake of proving APX-hardness of several geometric optimization problems. As Special-3VC is APX-hard we obtain the following theorem.

▶ **Theorem 21.** *Minimum constraint removal with rectangles is APX-hard.*

References

1 P. Agarwal, N. Kumar, S. Sintos, and S. Suri. Computing shortest paths in the plane with removable obstacles. In *16th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT 2018)*, pages 5:1–5:15, 2018.

2 H. Alt, S. Cabello, P. Giannopoulos, and C. Knauer. On some connection problems in straight-line segment arrangements. *27th EuroCG*, pages 27–30, 2011.

3 S. Bereg and D. G. Kirkpatrick. Approximating barrier resilience in wireless sensor networks. In *5th ALGOSENSORS 2009*, pages 29–40, 2009.

4 H. J. Broersma, X. Li, G. Woeginger, and S. Zhang. Paths and cycles in colored graphs. *Australasian journal of combinatorics*, 31(1):299–311, 2005.

5 D. Y. C. Chan and D. G. Kirkpatrick. Multi-path algorithms for minimum-colour path problems with applications to approximating barrier resilience. *Theor. Comput. Sci.*, 553:74–90, 2014.

6 T. M. Chan and E. Grant. Exact algorithms and apx-hardness results for geometric packing and covering problems. *Computational Geometry*, 47(2):112–124, 2014.

7 K.L. Clarkson and P.W. Shor. Application of random sampling in computational geometry, II. *Discrete & Computational Geometry*, 4:387–421, 1989.

8 I. Dinur and D. Steurer. Analytical approach to parallel repetition. In *Symposium on Theory of Computing, STOC 2014*, pages 624–633, 2014.

9 E. Eiben, J. Gemmell, I. Kanj, and A. Youngdahl. Improved results for minimum constraint removal. In *Proceedings of AAAI, AAAI press*, 2018.

10 E. Eiben and I. Kanj. How to navigate through obstacles? *CoRR*, abs/1712.04043, 2017. URL: `http://arxiv.org/abs/1712.04043`.

11 L. Erickson and S. LaValle. A simple, but np-hard, motion planning problem. In *Proceedings of AAAI, AAAI press*, 2013.

12 M. R. Fellows, J. Guo, and I. Kanj. The parameterized complexity of some minimum label problems. *Journal of Computer and System Sciences*, 76(8):727–740, 2010.

13 R. Hassin, J. Monnot, and D. Segev. Approximation algorithms and hardness results for labeled connectivity problems. *J. Comb. Optim.*, 14(4):437–453, 2007.

14 K. Hauser. The minimum constraint removal problem with three robotics applications. In *Tenth Workshop on the Algorithmic Foundations of Robotics, WAFR 2012*, pages 1–17, 2012.

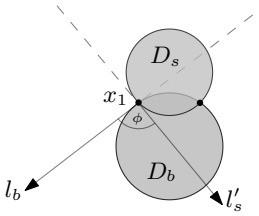


Figure 4 Angular Separation.

15 J. Hershberger, N. Kumar, and S. Suri. Shortest paths in the plane with obstacle violations. In *25th Annual European Symposium on Algorithms, (ESA 2017)*, pages 49:1–49:14, 2017.

16 K. Kedem, R. Livne, J. Pach, and M. Sharir. On the union of jordan regions and collision-free translational motion amidst polygonal obstacles. *Discrete & Computational Geometry*, 1:59–70, 1986.

17 M. Korman, M. Löffler, R. I. Silveira, and D. Strash. On the complexity of barrier resilience for fat regions. In *9th ALGOSENSORS 2013*, pages 201–216, 2013.

18 S.Khot and O. Regev. Vertex cover might be hard to approximate to within 2-epsilon. *Journal of Computer and System Sciences*, 74(3):335–349, 2008.

19 K-C. R. Tseng and D. G. Kirkpatrick. On barrier resilience of sensor networks. In *7th ALGOSENSORS 2011*, pages 130–144, 2011.

20 S. Yuan, S. Varma, and J. P. Jue. Minimum-color path problems for reliability in mesh networks. In *24th INFOCOM 2005*, volume 4, pages 2658–2669. IEEE, 2005.

A Proof of Lemma 16

To prove this lemma, we fix a small disk D_s and try to bound the number of bigger disks D_b intersected by D_s such that the tuple (x_1, D_s, D_b) was included in $\mathcal{P}$. To this end, we consider the following iterative procedure of adding tuples to $\mathcal{P}$ for a given D_s.

We process a set of *event points* $\mathcal{E}$ along the boundary of D_s in clockwise order. The event points are of the form $e_i = \langle x_1^i, D_s, D_i \rangle$ such that radius of D_i is at least the radius of D_s and D_i intersects D_s transversally at points x_1^i, x_2^i as defined before. (See also Figure 2(a).) Starting at any arbitrary point on the boundary of D_s, we sort events in $\mathcal{E}$ by its intersection point x_1^i in clockwise order. If two events $e_i, e_j \in \mathcal{E}$ have the same intersection point $x_1^i = x_1^j = x_1$, we order them by their *angular separation* from D_s at point x_1.

For any pair of intersecting disks D_s, D_b we define the angular separation at a point of intersection x_1 as follows. Consider the half line tangent l_b to disk D_b at x_1 such that l_b does not intersect D_s. Similarly, consider the half line tangent l'_s to disk D_s at x_1 but such that l'_s *intersects* D_b. The angle ϕ between l_b and l'_s is called the angular separation of D_b from D_s at the point x_1. (See also Figure 4).

Initially all disks D_i such that $e_i = \langle x_1^i, D_s, D_i \rangle$ is an event point in $\mathcal{E}$ are unmarked. Now, we simply process the events in $\mathcal{E}$ in the aforementioned order and mark the corresponding disks D_i as either *processed* or *ignored* or *discarded*. At iteration i, we consider the event point e_i and process the corresponding (unmarked) disk D_i. Now we consider the critical set $\mathcal{D}(D_s, D_i)$ as we defined before. If $|\mathcal{D}(D_s, D_i)|$ is at most k, we mark D_i as *processed* and all the unmarked disks in $\mathcal{D}(D_s, D_i)$ as *ignored*. Otherwise, we mark D_i as *discarded* and proceed to the next event point. For example, in Figure 5, we start at point x_1^1 and proceed in clockwise order. D_1 appears before D_2 in $\mathcal{E}$ because the angular separation with D_s is smaller for D_1. The disks marked as processed are shown with thick boundary and shaded gray. The disks that are marked ignored are shown in dotted and shaded orange.

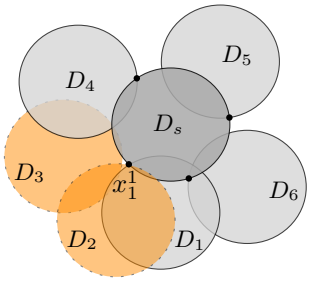


Figure 5 Processing disks D_i.

Observe that during each iteration we mark at most one disk as processed and at most k disks as ignored. Since we do not add tuples for disks that are discarded, we could have added at most $k+1$ tuples to $\mathcal{P}$ in each iteration. Next, we will show that the process terminates after a constant number of iterations. To see this, observe that the disk D_i that is marked as processed during iteration i must be exterior-disjoint w.r.t D_s from the disk D_{i-1} marked processed at the previous iteration. Thus among all these processed disks only the last disk can intersect the first one in the exterior of D_s. All other disks are exterior-disjoint of D_s from these two disks and themselves. Since all these disks are bigger than D_s, from Lemma 15 it follows that, we can have at most seven such disks around D_s. Therefore, the process must terminate after at most seven iterations and the number of tuples added to $\mathcal{P}$ corresponding to D_s is at most $7k+7$. Since there are $O(n)$ choices for D_s, we have in total $O(kn)$ candidates added to $\mathcal{P}$. ◀

B 2-Inapproximability for Rectilinear Polygons

We reduce an instance of Vertex Cover to an instance of MCR with rectilinear polygons. Recall that in the Vertex Cover problem we are given an n vertex graph $G=(V,E)$, and the goal is to find a minimum size subset $V' \subseteq V$ such that for any $(u,v) \in E$, either u or v is in V'. Let $e_1, \ldots, e_m$ be the edges of G. Next, we describe the reduction. The reduced instance of MCR contains a region called *barrier* formed by a subset of the obstacles. Each point in the barrier is contained in more than n obstacles and thus if an s-t path intersects the barrier, it intersects more than n obstacles. We would ensure that any optimal path of the instance intersects at most n obstacles and thus no such path intersects the barrier region. Intuitively, the barrier region forces any optimal path to lie in a certain region, which we refer to as *corridor*.

The construction is the following. We place an obstacle corresponding to each vertex. For each edge (u,v) there is two possible pathlets (or subpaths of an s-t path) - one that intersects the obstacle corresponding to u and the other that intersects the obstacle corresponding to v. The start (resp. end) points of the two pathlets corresponding to an edge are the same. Also one of the two pathlets corresponding to an edge lies above x-axis and the other lies below x-axis. To ensure this all the start and the end points of the pathlets are placed on the x-axis. Let s_i and t_i be the respective start and end points of the pathlets corresponding to the edge e_i. These points are placed on x-axis in the order $s_1, t_1, s_2, t_2, \ldots, s_m, t_m$. For each $1 \leq i < m$, we connect the point t_i with s_{i+1} using a segment that joins the i^{th} and $i+1^{th}$ pathlets. The point s is placed on x-axis before s_1 and t is placed on x-axis after t_m. s and s_1 are connected by a segment. Similarly, t_m and t are connected by a segment. Now to ensure that the pathlets cross the correct obstacles, they are laid out as shown in Figure 6.

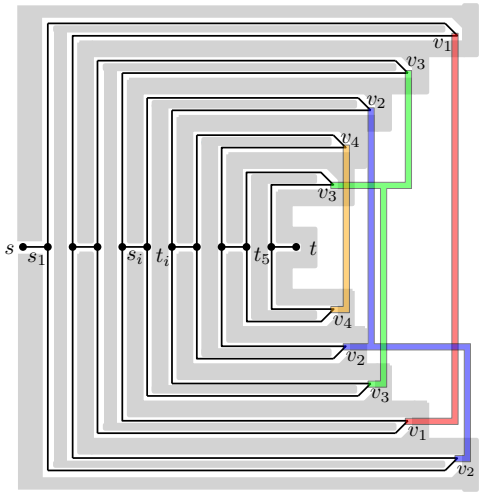


Figure 6 An example of the construction. The barrier region is shown in gray.

Each pathlet contains exactly one point (tip) having the maximum x-coordinate. Moreover, all such tips corresponding to the pathlets are in convex position. Thus one can connect the tips of any subset of pathlets using segments to form a rectilinear polygon that does not intersect any other pathlets (see Figure 6). Recall that each pathlet of an edge corresponds to a vertex. For each vertex $u \in V$, we connect the tips of the pathlets corresponding to u to form a rectilinear polygon shaped obstacle. Note that the total number of possible s-t paths we constructed is 2^m. Now to make sure that any optimal s-t path is one of these 2^m paths, we surround these paths with a barrier. Any optimal path must always stay inside the corridor, as it is expensive to cross the "wall" of the barrier. As the pathlets consist of a polynomial number of segments in total, a polynomial number of rectilinear polygons is sufficient to place avoiding the 2^m s-t paths. We make $O(n)$ copies of each such polygon to ensure the density. Lastly, each obstacle corresponding to a vertex is expanded sufficiently to ensure that it blocks the respective portion of the corridor. Note that the barrier can be placed in a way so that the corridor is arbitrarily thin, and thus this expansion can be done such that the obstacles do not cross any additional pathlets.

▶ **Lemma 22.** *There is a size k vertex cover for G iff there is an s-t path that intersects k obstacles.*

Proof. Given a cover $V' \subseteq V$ of size k for G, we simply remove the obstacles corresponding to vertices in G. Since V' covers all the edges of G, it must unblock at least one of the two pathlets corresponding to each edge (u, v) giving an obstacle-free path from s to t. Similarly given an s–t path π that intersects k obstacles, we can construct a cover for G by simply including the vertices corresponding to obstacles intersected by π. ◀

As Vertex Cover is hard to approximate within a factor of 2 assuming the Unique Games eonjecture [18], it follows that MCR with rectilinear polygons is hard to approximate within a factor of 2 (Theorem 19). It is easy to see that the same idea can easily be extended for convex polygons. Basically, one can connect the tips of any subset of pathlets using segments to form a convex polygon that does not intersect any other pathlets. This gives us the same hardness bound of 2 even for convex polygons (Corollary 20).

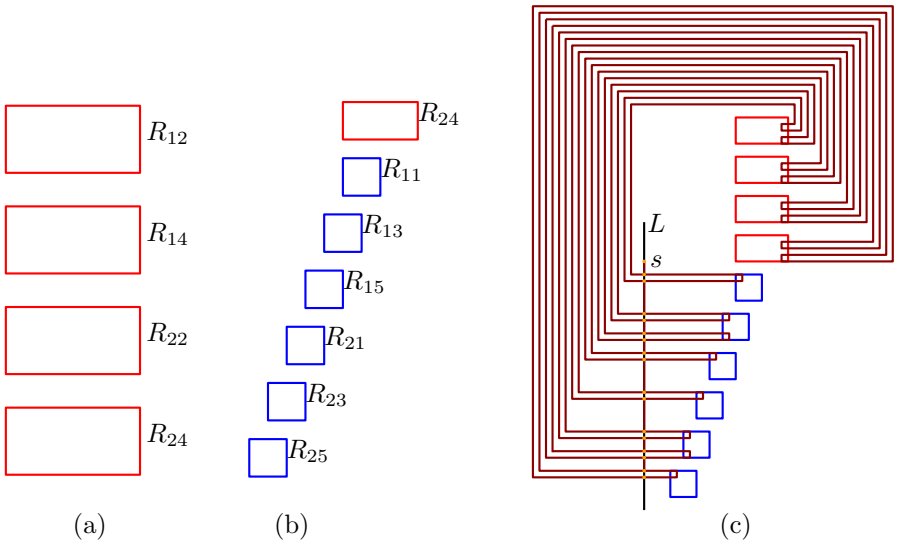


Figure 7 (a)The stack of the class 1 rectangles for $m = 2$. (b) The initial configuration of the class 2 rectangles (shown by squares) for $m = 2$. (c)Drawing of the pathlets for the class 1 edges.

C APX-hardness for Axis Parallel Rectangles

We reduce a restricted version of vertex cover to our problem which is referred to as Special-3VC. Chan et al. [6] had introduced this version for the sake of proving APX-hardness of several geometric optimization problems.

▶ **Definition 23.** In a Special-3VC instance, we are given a graph $G = (V, E)$, where V contains $5m$ vertices $\{v_{ij} \mid 1 \leq i \leq m, 1 \leq j \leq 5\}$. E contains $4m+n$ edges - $4m$ of type 1 and n of type 2, where $2n = 3m$. Type 1 edges are of the form $\{(v_{ij}, v_{i,j+1}) \mid 1 \leq i \leq m, 1 \leq j \leq 4\}$. Type 2 edges are of the form $\{(v_{pq}, v_{xy}) \mid 1 \leq p < x \leq m,$ and q, y are odd numbers$\}$ such that any vertex v_{ij} with odd index j appears in exactly one such edge.

As each vertex v_{ij} with odd index j contributes exactly once in the type 2 edges, the number of type 2 edges is $3m/2 = n$. Chan et al. [6] proved that Special-3VC is APX-hard. Now we describe our reduction. The reduction is similar to the reduction for rectilinear polygons. We will have one obstacle corresponding to each vertex. Moreover, we construct two pathlets corresponding to each edge (u, v) such that one pathlet intersects the obstacle corresponding to u and the other intersects the obstacle corresponding to v. However, due to the simpler structure of the obstacles, here it is more complicated to show that the pathlets intersect the correct obstacles. The construction of the instance of MCR is as follows.

We denote the rectangles corresponding to v_{ij} by R_{ij}. First we place the rectangles corresponding to the vertices in $\{v_{ij} \mid 1 \leq i \leq m$ and j is even$\}$ in a way so that they form a stack like structure (see Figure 7(a)). Also the rectangles are placed from top to bottom in the lexicographic order of the indexes (i, j): R_{ab} is considered before R_{cd} if $a < c$, and R_{a2} is considered before R_{a4}. We refer to these rectangles as the class 1 rectangles. Thereafter we place the rectangles corresponding to the remaining vertices. All these rectangles are placed in the increasing order of the sum of the indexes $i + j$. The first one is placed below R_{m4} (the last rectangle of the stack) in a way so that its left side is aligned with the left side of R_{m4}. Thereafter every rectangle is placed below the already placed ones and a little aligned towards the left w.r.t. the previous one (see Figure 7(b)). We refer to these rectangles as the

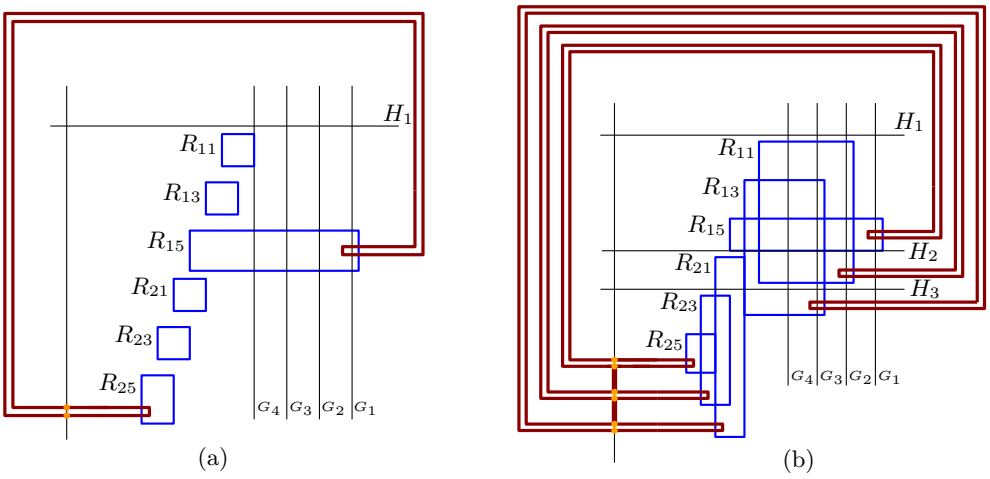


Figure 8 (a) Drawing of the pathlets for the edge (v_{pq}, v_{m5}) where $m = 2, p = 1, q = 5$. (b) Drawing of the pathlets for the type 2 edges $(v_{15}, v_{25}), (v_{11}, v_{23}), (v_{13}, v_{21})$ where $m = 2$.

class 2 rectangles. We note that initially every class 2 rectangle is a square. Later each such rectangle might be expanded suitably towards right and below to ensure the correctness of the intersections with the pathlets.

Now let L be a vertical line such that all the rectangles are placed strictly to the right of it. All the endpoints of the pathlets we draw lie on L. Each pathlet is a curve consisting of rectilinear segments. The start (resp. end) points of the two pathlets corresponding to an edge are the same. We place s right above the topmost start point of the pathlets and connect s with this point by a vertical segment. Similarly, the point t is placed below the bottommost end point and joined with it by a vertical segment. At first we draw the pathlets for type 1 edges $\{(v_{ij}, v_{i,j+1}) \mid 1 \leq i \leq m, 1 \leq j \leq 4\}$ in the dictionary order of the indexes $(i, j, i, j+1)$, i.e at first (v_{11}, v_{12}), then (v_{12}, v_{13}) and so on. The pairs of start and end points of the pathlets corresponding to these edges appear in the same order on L from top to bottom. For each type 1 edge $(v_{ij}, v_{i,j+1})$, let $s(i, j, j+1)$ and $t(i, j, j+1)$ be the respective start and end points of the pathlets. j is 3. Otherwise, it appears only once. Let P_{ij} be the horizontal projection (an interval) of R_{ij} on L. Then the start and endpoints of the pathlets of the type 1 edges with a vertex v_{ij} lie on P_{ij}. Now consider a type 1 edge $(v_{ij}, v_{i,j+1})$. Then either j or $j+1$ is odd. WLOG let j is odd. We draw the two points $s(i, j, j+1)$ and $t(i, j, j+1)$ on P_{ij} such that $s(i, j, j+1)$ lies above $t(i, j, j+1)$. One pathlet of $(v_{ij}, v_{i,j+1})$ lies on the right of L. It consists of three orthogonal segments and the only rectangle it intersects is R_{ij} (see Figure 7(c)). The other pathlet is also drawn in a way so that the only rectangle it intersects is $R_{i,j+1}$ (see Figure 7(c)). We repeat the process for all type 1 edges and each consecutive pairs of end and start points are joined with a vertical segment.

Now we draw the pathlets corresponding to the type 2 edges $\{(v_{pq}, v_{xy}) \mid 1 \leq p < x \leq m,$ and q, y are odd numbers$\}$. Note that, there are n such edges in G. We process all these edges in the reverse lexicographic order of the indexes (x, y) of the vertices v_{xy}. Thus at first we consider the edge that contains v_{m5}, then the edge that contains v_{m3} (if not considered already), then the edge that contains $v_{m-1,5}$ (if not considered already), and so on. We take $n+1$ vertical lines $G_1, \ldots, G_{n+1}$ such that G_{n+1} intersects the right vertical side of R_{11}, G_n is on the right of G_{n+1}, G_{n-1} is on the right of G_n, and in general G_i is on the right of G_{i+1}. Also let G_i and G_{i+1} are unit distance apart for $1 \leq i \leq n$. In every iteration $1 \leq i \leq n$, we define a horizontal line H_i. Denote the region by Q_i, that lie below H_i and inside the

strip defined by G_i and G_{i+1}. The drawing procedure is the following. Consider the first edge (v_{pq}, v_{m5}) corresponding to the vertex v_{m5}. Let H_1 be a horizontal line such that all the class 1 rectangles lie above it and all the class 2 rectangles lie below it. At first we expand R_{m5} sufficiently towards below such that one can place a pathlet with the following properties - the only rectangle it intersects is R_{m5}, it consists of two horizontal segments and one vertical segment, and its start and end points lie on L. Note that the expansion of R_{m5} do not create any new intersections with the existing pathlets. Thereafter R_{pq} is expanded sufficiently towards below and right to ensure that it has non-empty intersection with Q_1. Then the other pathlet can be drawn in a way so that it intersects the portion of R_{pq} that is in Q_1, and as Q_1 is empty the pathlet does not intersect any other rectangle (see Figure 8(a)). Now consider the i^{th} type 2 edge (v_{pq}, v_{xy}) in this order. Let all the edges before it in the ordering are already taken care of. It is easy to see that one can expand R_{xy} towards below for drawing a pathlet with the desired properties. Now to make sure that the other pathlet intersects only R_{pq}, set H_i to be a horizontal line such that the region Q_i, as defined above, is empty of previously drawn pathlets and expanded rectangles. Then we can expand R_{pq} towards below and right so that it has non-empty intersection with Q_i. As Q_i is empty one can draw the other pathlet as well with the desired properties (see Figure 8(b)).

Finally, we place the barrier region around the paths. As the pathlets are orthogonal and consisting of a polynomial number of segments in total, the barrier region can be simulated using a polynomial number of rectangles and thus the construction can be realized in polynomial time.

From the construction, it is straightforward to see the following lemma.

▶ **Lemma 24.** *There is a size k vertex cover for G iff there is an s-t path that intersects k rectangles.*

As Special-3VC is APX-hard, it follows that MCR with axis-aligned rectangles is APX-hard (Theorem 21).

A Tight 4/3 Approximation for Capacitated Vehicle Routing in Trees

Amariah Becker[1]
Brown University Department of Computer Science, Providence, RI, USA
amariah_becker@brown.edu

Abstract

Given a set of clients with demands, the Capacitated Vehicle Routing problem is to find a set of tours that collectively cover all client demand, such that the capacity of each vehicle is not exceeded and such that the sum of the tour lengths is minimized. In this paper, we provide a 4/3-approximation algorithm for Capacitated Vehicle Routing on trees, improving over the previous best-known approximation ratio of $(\sqrt{41}-1)/4$ by Asano et al.[2], while using the same lower bound. Asano et al. show that there exist instances whose optimal cost is 4/3 times this lower bound. Notably, our 4/3 approximation ratio is therefore tight for this lower bound, achieving the best-possible performance.

2012 ACM Subject Classification Theory of computation → Routing and network design problems

Keywords and phrases Approximation algorithms, Graph algorithms, Capacitated vehicle routing

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.3

1 Intro

Vehicle-routing problems address how a service can best be provided to meet the demand from a set of clients. These problems arise very naturally in both commercial and public planning. Real world vehicle-routing problems must account for the capacities of the vehicles, which limit the amount of client demand that can be met in a single trip. We formalize this problem as finding tours in a graph:

Given a graph $G = (V, E)$, a specified *depot* vertex r, a *client set* S, an edge length function $l : E \to \mathbb{Z}^{\geq 0}$, a demand function $d : S \to \mathbb{Z}^{\geq 0}$ and $Q > 0$, the Capacitated Vehicle Routing problem is to find a set of tours of minimum total length such that each tour includes r and the tours collectively *cover* all demand at every client and such that no tour covers more than Q demand. A tour can only cover demand from clients along the tour, but it may pass by some clients without covering their demand.

There are two common variants of this problem: *splittable* and *unsplittable*. In the splittable variant, the demand of a client can be collectively covered by multiple tours, and in the unsplittable variant, the entire demand of a client must be covered by the same tour.

Both variants of this problem are NP-hard and therefore unlikely to admit a polynomial-time exact solution, but constant factor approximations can be found in polynomial time [9]. A natural question is whether better performance can be achieved for restricted graph classes. One line of research has focused on approximating Capacitated Vehicle Routing in trees. Though the problem remains NP-hard in trees [12], better constant-factor approximations have been found than for general metrics.

[1] Research funded by NSF grant CCF-14-09520

Hamaguchi and Katoh [10] noted a simple lower bound for the splittable-variant of Capacitated Vehicle Routing in trees: every edge must be traversed by at least enough vehicles to accommodate all demand from the clients that use the edge on the shortest path to the depot. They then use this lower bound (denoted LB) to give a 1.5 approximation [10]. Following this work, Asano et al. [2] use the same lower bound to achieve a $(\sqrt{41}-1)/4$ approximation. They also prove the following lemma (see Appendix C):

▶ **Lemma 1** ([2]). *There exist instances of* Capacitated Vehicle Routing *in trees whose optimal solution costs* $4/3 \cdot LB$.

This shows that the best possible approximation ratio using this lower bound would be a 4/3-approximation. Our result, stated in Theorem 2, achieves this ratio, and is therefore tight with respect to LB. No further improvements over our result can be made until a better lower bound is found.

▶ **Theorem 2.** *There is a polynomial-time 4/3 approximation for* Capacitated Vehicle Routing *in trees that is tight with respect to* LB.

1.1 Related Work

As Capacitated Vehicle Routing generalizes the Traveling Salesman Problem (TSP) (which is the special case of $Q = |V|$) it is NP-hard, and in general metrics is APX-hard [13].

For general metrics, a technique called *Iterated Tour Partitioning* starts with a TSP solution, partitions this tour into paths of bounded capacity, and then makes vehicle routing tours by adding paths from the depot to each endpoint of each path [9]. Iterated Tour Partitioning results in a polynomial time $(1 + (1 - 1/Q)\alpha)$-approximation for splittable Capacitated Vehicle Routing, where α is the approximation ratio of the TSP tour. A similar approach can be used for the unsplittable variant, resulting in a $(2 + (1 - 2/Q)\alpha)$-approximation [1]. Using Christofides' 1.5-approximation for TSP [6], these ratios are $(2.5 - \frac{1.5}{Q})$ and $(3.5 - \frac{3}{Q})$ respectively. No significant improvements over iterated tour partitioning are known for general metrics.

Even in trees, splittable Capacitated Vehicle Routing is NP-hard by a reduction from bin packing [12], and unsplittable Capacitated Vehicle Routing is NP-hard to even approximate to better than a 1.5-factor [8]. Since depth-first search trivially solves TSP optimally in trees, iterated tour partitioning already gives a $(2 - \frac{1}{Q})$-approximation for splittable demands in trees and a $(3 - \frac{2}{Q})$-approximation for unsplittable demands in trees. Labbe et al. improved this to a 2-approximation for the unsplittable-demand variant [12]. For splittable Capacitated Vehicle Routing in trees, Hamaguchi and Katoh [10] define a natural lower bound on the cost of the optimal solution and give a 1.5-approximation algorithm that yields a solution with cost at most 1.5 times this lower bound. Asano, Katoh, and Kawashima [2] improve the ratio to $(\sqrt{41}-1)/4$ using this same lower bound. They also show that there are instances whose optimal cost is, asymptotically, 4/3 times this lower bound value (see Appendix C for such an instance). This implies that if a 4/3-approximation algorithm exists that uses this lower bound it would be *tight* (i.e. best possible). In this paper we resolve the question as to whether or not such an an algorithm exists.

All of the above results allow for arbitrary capacity Q. Even for fixed capacity $Q \geq 3$, Capacitated Vehicle Routing is APX-hard in general metrics [3]. For fixed capacities, Capacitated Vehicle Routing is polynomial-time solvable in trees, but is NP-hard in other metrics. For instances in the Euclidean plane ($\mathbb{R}^2$), polynomial-time approximation

schemes (PTAS) are known for instances where Q is constant [9], Q is $O(\log n / \log\log n)$ [3], and Q is $\Omega(n)$ [3]. For higher-dimensional Euclidean spaces $\mathbb{R}^d$, a PTAS is known for when Q is $O(\log^{1/d} n)$ [11]. While a *quasi*-polynomial-time approximation (QPTAS) is known for arbitrary Q on instances in the Euclidean plane ($\mathbb{R}^2$) [7], no PTAS is known for arbitrary Q in *any* non-trivial metric.

Recently, the first approximation schemes for non-Euclidean metrics were designed. Specifically, a quasi-polynomial time approximation scheme is known for planar and bounded-genus graphs when Q is fixed [5], and a polynomial-time approximation scheme is known for graphs of bounded highway-dimension when Q is fixed [4].

1.2 Techniques

Our work extends the techniques introduced by Asano et al. [2] which itself was an extension of the work of Hamaguchi and Katoh [10]. Specifically, Hamaguchi and Katoh describe a very natural lower bound that arises on tree instances [10]: the number of tours that traverse each edge must be at least enough to cover all demand in the subtree below the edge (the tree is assumed to be rooted at the depot). This introduces a minimum *traffic* value on the edge. Multiplying this value by two (each tour crosses each edge once in each direction) times the weight of the edge and summing over all edges provides a lower bound on the cost of any feasible solution.

The algorithm of Asano et al. [2] proceeds in a sequence of rounds. In each round, a set of tours is identified such that the *cost-to-savings ratio*, namely the ratio of the cost (of these tours) to the reduction to the lower bound that results from taking these tours, is bounded by some constant α. The key is that although a given tour itself may cost more than α times its reduction to the lower bound, collectively the set of tours has the desired ratio. If after a round ends, no uncovered demand remains, then the union of these sets of tours is a feasible solution with cost at most α times the lower bound, which is at most α times the optimal cost. For Asano et al. [2], $\alpha = (\sqrt{41} - 1)/4$. We use a similar approach to achieve $\alpha = 4/3$.

Asano et al. [2] also introduced the idea of making *safe* modifications to the instance. That is, modifying the structure of the instance in such a way that does not increase the value of the lower bound or decrease the optimum cost (although it may increase the optimum cost) and such that a feasible solution in the modified instance has a corresponding feasible solution in the original instance. These modifications can be made safely at any point in the algorithm. Our algorithm also makes use of safe modifications, although the ones that we define differ somewhat from those defined by Asano et al. [2].

These modifications allow us to reason better about how the resulting instance must be structured. The idea is that the modifications can be made until one of a few cases arise. Each case has a corresponding *strategy* to find a set of tours with the desired cost-to-savings ratio.

Specifically, the algorithm of Asano et al. [2] classifies the *leafmost* subtrees containing at least $2Q$ units of demand into one of a few cases. The main obstacle in extending their algorithm to a 4/3-approximation is that one of the cases does not seem to have a good strategy. On the other hand, modifying the algorithm to instead classify the *leafmost* subtrees containing at least βQ units of demand for some $\beta > 2$ can greatly increase the number of cases that arise.

We overcome this obstacle by generalizing the difficult case into what we call a *p-chain* (See Figure 2). Our key insight is that even arbitrarily large p-chains can be addressed efficiently in sibling pairs and at the root. Our algorithm effectively delays addressing the difficult case by pushing it rootward until it finds a pair or reaches the root and is thus easy

to address. To keep the number of cases small, we address easy cases as they emerge, and require that any remaining difficult case must have a specific structure that the algorithm can easily detect in subsequent rounds.

2 Preliminaries

We use OPT to denote the cost of an optimization problem. For a minimization problem, a polynomial-time α-approximation algorithm is an algorithm with a runtime that is polynomial in the size of the input and returns a feasible solution with cost at most $\alpha \cdot OPT$.

When the input graph G is a tree it is assumed to be rooted at the depot r. Let $P[u, v]$ denote the unique path from u to v in the tree. Recall that the problem gives a length $l(e) \geq 0$ for each edge e, and let $l(P[u, v]) = \sum_{e \in P[u,v]} l(e)$ denote the shortest path distance between u and v.

For rooted tree $T = (V, E)$, and $v \in V$ let T_v denote the subtree rooted at v and $d(T_v)$ be the total demand from vertices in T_v.

The parent of a vertex v is the vertex u adjacent to v in $P[v, r]$, and the parent of r is undefined. An edge labeled (u, v) indicates that u is the parent of v. The parent of an edge (u, v) is the edge $(parent(u), u)$ and is undefined if $u = r$. If v has parent vertex u, we call $B = T_v \cup \{(u, v)\}$ a *branch* at u, and edge (u, v) the *stem* of the branch. The parent of a branch at u with stem e is the branch at $parent(u)$ with stem $parent(e)$ and is undefined if $u = r$. A vertex (resp. edge, branch) with a parent is said to be a child vertex (resp. edge, branch) of that parent.

2.1 Lower Bound

Here we describe the lower bound introduced by Hamaguchi and Katoh [10] and Asano et al. [2] and adopt similar terminology. Since all demand must be covered and each tour can cover at most Q demand, each edge $e = (u, v)$ must be traversed by enough tours to cover $d(T_v)$ demand. We call this value the *traffic* on the edge e, denoted $f(e)$. Namely, $f(e) = \lceil \frac{d(T_v)}{Q} \rceil$ tours. Each such tour traverses the edge exactly twice (once in each direction). We say that the lower bound $LB(e)$ of the contribution of edge $e = (u, v)$ to the total solution cost is therefore,

$$LB(e) = 2 \cdot l(e) \cdot f(e) = 2 \cdot l(e) \lceil \frac{d(T_v)}{Q} \rceil$$

and that the lower bound LB on OPT is

$$LB = \sum_{e \in E} LB(e)$$

For convenience, we scale down all demand values by a factor of Q and set $Q = 1$. We also assume that the vertices with positive demand are exactly the leaves: If some internal vertex v has positive demand $d(v)$ we can add a vertex v' with demand $d(v') = d(v)$ and edge (v, v') of length zero and set $d(v)$ to zero. Alternatively, if some leaf v has zero demand, no tour in an optimal solution will visit v, so v and the edge to v's parent can be deleted from the graph. Finally, we assume that no non-root vertex has degree exactly two, as no branching would occur at such a vertex, so the two incident edges can be spliced into one.

A very high level description of the algorithm is as follows: iteratively identify sets of tours in which the ratio of the cost of the tours to the reduction in cost to the lower bound, LB, is at most 4/3. We call such a set of tours a 4/3-*approximate tour set*.

We say that a 4/3-approximate tour set *removes* the demand that the tours cover. If such a tour set removes all demand from a branch, we say that this branch has been *resolved*. After a branch is resolved, it is convenient to think of it as having been deleted from the tree and proceed with the smaller instance.

We note that any 4/3-approximation algorithm for Capacitated Vehicle Routing trivially generates a 4/3-approximate tour set. The converse, is also straightforward:

▶ **Lemma 3.** *If, after iteratively finding and removing demand from 4/3-approximate tour sets, no demand remains, then the union of all tour sets is a 4/3-approximation for* Capacitated Vehicle Routing.

Given this, we make one more simplifying assumption that each leaf v has demand $d(v) < 1$. Assume to the contrary that for some leaf v, $d(v) \geq 1$. A tour that goes directly from r to v and back and covers one unit of demand at v is in fact a 1-approximate tour (and thus also a 4/3-approximate tour). If ever such a leaf exists, we can greedily take such tours until no more such leaves exist [2].

2.2 Safe Operations

We say that an operation that modifies the graph is *safe* if it does not decrease OPT or change the cost of the lower bound LB and it preserves feasibility. Note that a 4/3-approximate tour set in the modified graph is therefore also a 4/3-approximate tour set in the unmodified graph.

The algorithm proceeds iteratively. In each iteration, the algorithm performs a series of *safe* operations and takes a sequence of 4/3-approximate tours.

We now define the operations (see Appendix A for proofs of safeness). Note that our operations and simplifying assumptions generalize the operations that appear in [2] with different vocabulary (see Appendix B for a comparison).

- **Condense**: If edge $e = (u, v)$ has traffic $f(e) = 1$ and v is not a leaf, add a vertex v' and replace e and T_v with an edge $e' = (u, v')$ with $l(e') = l(e) + \sum_{e'' \in T_v} l(e'')$ and set $d(v') = d(T_v)$ (see Figure 1a).
- **Unzip**: If edge $e = (u, v)$ has traffic equal to the sum of the traffic on child edges $(v, w_1), (v, w_2), ..., (v, w_k)$, then delete v and add edges $(u, w_1), (u, w_2), ..., (u, w_k)$ with lengths $l((u, w_i)) = l(e) + l(v, w_i)$ for all $i \in \{1, 2, ..., k\}$ (see Figure 1b).
- **Group**: If vertex u has at least four children, including three leaf children v_1, v_2, v_3, such that $1.5 < d(v_1) + d(v_2) + d(v_3) < 2$, add vertex u', edge (u, u') of length zero, and for $i \in 1, 2, 3$, replace (u, v_i) with (u', v_i) (see Figure 1c).
- **Unite**: If vertex u has leaf children v_1 and v_2 such that $d(v_1) + d(v_2) \leq 1$, delete v_1 and v_2 and add vertex v_0 with demand $d(v_1) + d(v_2)$ and edge (u, v_0) with length $l((u, v_1)) + l((u, v_2))$ (see Figure 1d).
- **Slide**: If edge $e_0 = (u, v)$ has child edges $e_1 = (v, w_1)$ and $e_2 = (v, w_2)$ such that traffic $f(e_0) = f(e_1)$, then delete edge e_2 and add edge (w_1, w_2) of length $l(e_2)$ (see Figure 1e).

3 Algorithm

Exhaustively applying safe operations is called *simplifying* the instance. Note that none of the operations cancel each other, so this process terminates.

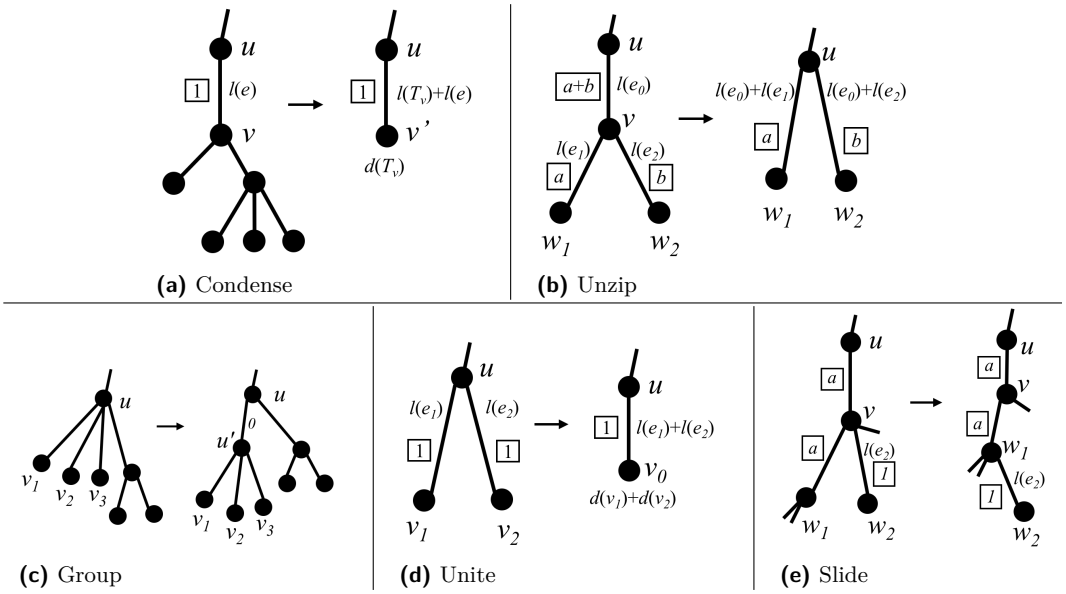


Figure 1 Safe Operations. Traffic values are shown in rectangles.

We say that a problem instance is *simplified* if no more safe operations are available, no internal vertices have demand, no non-root vertex has degree two, and for every leaf v, $0 < d(v) < 1$. We say that a branch is simplified if these conditions hold for the branch.

Recall that a *branch* consists of a subtree along with its parent edge (*stem*). If the branch has traffic p, we call it a *p-branch*.

A simplified 2-branch with stem $e_0 = (u, v)$ such that v has exactly three children w_1, w_2, w_3, all of which are leafs and such that $1.5 < d(w_1) + d(w_2) + d(w_3) \leq 2$ is called a 2-*chain*.

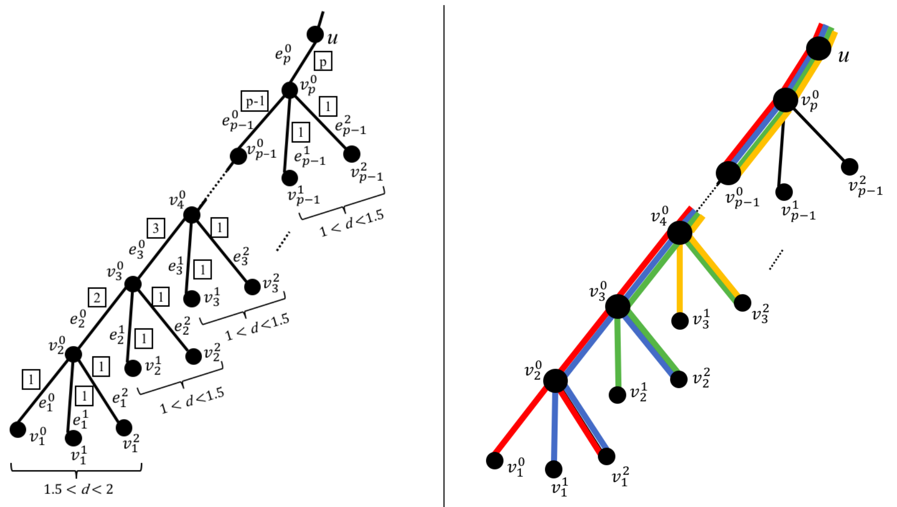
▶ **Lemma 4.** *In a simplified problem instance, all 2-branches are 2-chains.*

Proof. Consider any 2-branch in a simplified problem instance. Clearly no edge in the branch can have traffic greater than two. If more than one child edge of the stem had traffic two, then the traffic of the stem itself must be greater than two. If the stem had exactly one child edge with traffic two, then a slide operation would have been possible. Therefore every child edge of the stem has traffic one. Each of these edges must be leaf edges or else they could be condensed. If there were exactly two such edges, then the stem could be unzipped. Since no two of these edges can be united, then the demand of every pair sums to more than one, so there are exactly three such edges and their demand sums to more than 1.5. ◀

3.1 p-Chains

We now generalize the notion of a 2-chain. For $p \geq 3$ a *p-chain* is a simplified p-branch with stem $e_0 = (u, v)$ such that v has exactly three children w_1, w_2, w_3, in which w_2 and w_3 are leaves with $1 < d(w_2) + d(w_3) \leq 1.5$ and (v, w_1) is the stem of a $(p-1)$-chain (see Figure 2a).

For convenience, we define a labeling scheme for p-chains. Each vertex v_i^j is doubly indexed by level i and rank (child-order) j. We use e_i^j to denote the parent edge of v_i^j, $\forall i, j$. A 2-chain with parent u has stem $e_2^0 = (u, v_2^0)$ leaves v_1^0, v_1^1, and v_1^2 such that $l(e_1^0) \geq l(e_1^1) \geq l(e_1^2)$. For $p > 2$, a p-chain with parent u has stem $e_p^0 = (u, v_p^0)$, and children v_{p-1}^0, v_{p-1}^1, and v_{p-1}^2, such that e_{p-1}^0 is the stem of a $p-1$-chain, and v_{p-1}^1 and v_{p-1}^2 are leaves with $l(e_{p-1}^1) \geq l(e_{p-1}^2)$ (See Figure 2a).



(a) p-Chain. Traffic values are shown in rectangles, and d denotes demand.

(b) Cascade. The first four tours of a cascade tour set are depicted.

Figure 2

We further classify some p-chains as *long* p-chains: All 2-chains are long, and for $p \geq 3$ a *long* p-*chain* is a p-chain in which $l(e^2_{p-1}) < l(P[v^0_p, r])$ and e^0_{p-1} is the stem of a *long* $(p-1)$-chain. A p-chain in which $l(e^2_{p-1}) \geq l(P[v^0_p, r])$ is called a *short* p-chain.

Long p-chains are particularly convenient because they can be *resolved* individually at the root and in sibling pairs for internal vertices, as described in the following lemmas (which we prove in Section 5).

▶ **Lemma 5.** *Long p-chains can be resolved at the root.*

▶ **Lemma 6.** *A long p-chain and long p'-chain can be resolved together if they are sibling branches.*

As in [2], our algorithm proceeds in a series of iterations. Each iteration performs a set of safe operations and identifies a 4/3-approximate tour set.

▶ **Lemma 7.** *Iteration i runs in polynomial time and either finds a nonempty 4/3-approximate tour set or finds that every branch at the root is either a long p-chain or 1-branch.*

Proof. See Section 4. ◀

These iterations continue until every branch at the root is either a long p-chain or 1-branch. The algorithm then solves the remaining instance, using the result from Lemma 8.

▶ **Lemma 8.** *There is a polynomial time 4/3-approximation algorithm for instances of* Capacitated Vehicle Routing *on trees in which every child branch of the root is either a long p-chain or a 1-branch.*

Proof. The cost of a tour that traverses a 1-branch at the root is equivalent to the reduction to lower bound LB that results from removing the demand of the branch, so such a tour is a 1-approximate tour (and thus also a 4/3-approximate tour). The lemma result then follows from Lemma 3 and Lemma 5 ◀

Putting these steps together, gives our overall 4/3-approximation result described in Theorem 2.

▶ **Theorem 2.** *There is a polynomial-time 4/3 approximation for* Capacitated Vehicle Routing *in trees that is tight with respect to LB.*

Proof. Let m denote the total amount of demand in the graph. Since each iteration removes demand, there are at most m iterations, each of which runs in polynomial time. By Lemma 7, after iteration m, every branch off the root must be a long p-chain or a 1-branch. By Lemma 8 there is a polynomial-time 4/3-approximation for these branches. Combining this approximation with the collection of tours identified during the iterations results in an overall 4/3 approximation, by Lemma 3. ◀

All that remains is to prove the above lemmas. In Section 4 we describe the iteration subroutine and prove Lemma 7. In Section 5 we prove Lemmas 5 and 6.

4 Description of Iteration i

We say that a p-branch B is *settled* if it is either a 1-branch or a long p-chain. Otherwise we say that B is *unsettled*. We say that a p-branch B is *minimally unsettled* if it is unsettled and all of its child branches are settled.

Each iteration consists of the following subroutine:

1. Simplify the instance (i.e. exhaustively apply safe operations)
2. If all branches at the root are settled, terminate iteration.
3. Otherwise, find a minimally unsettled branch B.
   - **(i)** If B has at least two child branches that are long p-chains, apply Lemma 6.
   - **(ii)** Otherwise, if B has at least three child branches that are 1-branches, apply Lemma 9
   - **(iii)** Otherwise, B is a (short) p-chain. Apply Lemma 10.

▶ **Lemma 9.** *A simplified, minimally unsettled branch B with at least three child branches that are 1-branches admits a 4/3-approximate tour set.*

Proof. Let (u, v_0) be the stem of B, and let (v_0, v_1), (v_0, v_2), and (v_0, v_3) be three (child) 1-branches. Since the branch is simplified, v_1, v_2, and v_3 are leaves. Since the unite operation is unavailable, $d(v_1) + d(v_2) + d(v_3) > 1.5$, and since B is unsettled, it is not a 2-chain. Furthermore, the group operation is unavailable, so $2 < d(v_1) + d(v_2) + d(v_3) < 3$. Let $a = l(P[v_0, r])$, $w_1 = l(v_0, v_1)$, $w_2 = l(v_0, v_2)$, and $w_3 = l(v_0, v_3)$. Without loss of generality, assume $w_1 \leq w_2 \leq w_3$.

If $a \leq w_1 + w_2 + w_3$, then for $i \in \{1, 2, 3\}$, let t_i be the tour that travels from the depot to v_i, covers all demand at v_i, and then returns to the depot. The length of t_i is $2(a + w_i)$. The total cost of the tour set $\{t_1, t_2, t_3\}$ is $2(3a + w_1 + w_2 + w_3)$. This tour set covers all demand of the leaves and also reduces demand along $P[v_0, r]$ by two, since $2 < d(v_1) + d(v_2) + d(v_3) < 3$, so the reduction to LB is $2(2a + w_1 + w_2 + w_3)$. The ratio of the cost of the tour set to the reduction to the cost of LB that results from taking these tours is therefore,

$$\frac{2(3a + w_1 + w_2 + w_3)}{2(2a + w_1 + w_2 + w_3)} = 1 + \frac{a}{2a + w_1 + w_2 + w_3} \leq \frac{4}{3}$$

where the final equality comes from $a \leq w_1 + w_2 + w_3$.

Otherwise, $a > w_1 + w_2 + w_3$. Let t be the tour that travels from the depot to v_1, covers all demand at v_1, travels to v_3, covers as much demand as possible at v_3, and then returns to the depot. The cost of t is $2(a + w_1 + w_3)$. Note that since the unite operation is unavailable, then $d(v_1) + d(v_3) > 1$, so the vehicle is full, and some demand remains at v_3. Since the vehicle is full, then t reduces demand along $P[v_0, r]$ by one, so the reduction to

LB is $2(w_1 + a)$. The ratio of the cost of the tour set $\{t\}$ to the reduction to the cost of LB that results from taking these tours is therefore,

$$\frac{2(a + w_1 + w_3)}{2(w_1 + a)} = 1 + \frac{w_3}{w_1 + a} \leq \frac{4}{3}$$

where the final inequality comes from $a > w_1 + w_2 + w_3 \geq 3w_3$. ◀

▶ **Lemma 10.** *A short p-chain admits a 4/3-approximate tour set.*

Proof. Let $a = l(P[v_p^0, r])$, $b = l(e_{p-1}^2)$, and $c = l(e_{p-1}^1)$. By construction, $c \geq b$, and since the p-chain is short, $b \geq a$. Let t_1 be the tour that goes from the depot to v_{p-1}^1, covers all demand at this leaf, and then returns to the depot. Similarly, let t_2 be the tour that goes from the depot to v_{p-1}^2, covers all demand at this leaf, and then returns to the depot. Tour t_1 has cost $2(a + c)$ and t_2 has cost $2(a + b)$. Tour set $\{t_1, t_2\}$ has total cost $2(2a + b + c)$. This tour set covers all demand at these leaves, which sum to greater than one by definition of p-chain, so these tours reduce the demand along $P[v_p^0, r]$ by one. Therefore the reduction to LB from this tour set is $2(a + b + c)$. Therefore the ratio of the cost of the tour set to the reduction to the cost of LB that results from taking these tours is,

$$\frac{2(2a + b + c)}{2(a + b + c)} = 1 + \frac{a}{a + b + c} \leq \frac{4}{3}$$

The final inequality comes from $a \leq b \leq c$. ◀

Finally, we prove Lemma 7, which we restate here for convenience.

▶ **Lemma 7.** *Iteration i runs in polynomial time and either finds a nonempty 4/3-approximate tour set or finds that every branch at the root is either a long p-chain or 1-branch.*

Proof. To prove this lemma, we first must show that any minimally unsettled branch B in a simplified instance must be in at least one of the three identified cases. Let B be such a branch. B must have child branches, since it is unsettled. Since it is minimally unsettled, every child branch is settled (i.e. either a 1-branch or a long p-chain).

If B has at least two long p-chains as child branches, then case i holds.

If B has no long p-chains as child branches, then all child branches are 1-branches. By Lemma 4, in a simplified instance all 2-branches are 2-chains. Since all 2-chains are long, and thus settled, B must be a j-branch for some $j \geq 3$. Since the unzip operation is unavailable, then there are at least four 1-branches as child branches of B, so case ii holds.

If B has exactly one long p-chain as a child branch and case ii doesn't hold, then B must be a short p'-chain. If there were no 1-branches as child branches, then the degree-two vertex could be spliced. If there were exactly one 1-branch then either $p = p'$ and a slide operation would be available, or $p = p' - 1$ and an unzip operation would be available. Since case ii doesn't hold, there are exactly two 1-branches as child branches. Since an unzip operation is unavailable, $p = p' - 1$, and since a unite operation is unavailable, the demand of the two 1-branches (which are both leaves by the condense operation) sum to greater than one. Since the instance is simplified, then B is a p'-chain. But since B is unsettled, it must be a short p'-chain. Therefore case iii holds.

Since B is covered by some case, then the corresponding lemma (Lemma 6, 9, or 10) guarantees a nonempty 4/3-approximate tour set.

If no unsettled branch can be found, then all branches at the root must be settled.

Finally, to see that the iteration runs in polynomial time, note that each operation can be performed in constant time. Condense, unite, unzip, (and splicing) all make the

graph smaller, so they are exhausted in linear time. Subsequently, if condense and unite are unavailable, then each slide results in a degree two vertex remaining to be spliced, also making the graph smaller. After all other operations are exhausted, each leaf can participate in at most one group operation in a given iteration. Each of the three cases can likewise be handled in linear time. ◀

5 Resolving p-Chains

5.1 Cascades

In order to prove Lemmas 5 and 6 we describe a specific group of tours that collectively covers the demand of a long p-chain.

The labeling on p-chains gives a natural bottom-up ordering on the leaves: $v_1^0, v_1^1, v_1^2, v_2^1, v_2^2$, ..., v_{p-1}^1, v_{p-1}^2, where the order is determined first by level and then by rank. For a long p-chain, we define a *cascade* to be the sequence of p tours in which each tour considers the leaves in this bottom-up order and:

1. Visits and covers all demand from the first leaf with remaining demand.
2. While the vehicle has spare capacity and there is unmet demand in the branch, determines the lowest level i with leaves with unmet demand and covers as much demand as possible from v_i^2.
3. Returns to the depot.

Let the resulting tours be $t_1, ..., t_p$. Tour t_1 first covers all demand from v_1^0 and then covers $1 - d(v_1^0)$ demand from v_1^2. Since by definition, a p-chain is simplified, the unite operation is unavailable, so $d(v_1^0) + d(v_1^2) > 1$, implying the vehicle is full. The second tour t_2 covers all demand from v_1^1, covers all remaining demand at v_1^2 (since $d(v_1^0)+d(v_1^1)+d(v_1^2) < 2$) and covers some demand at v_2^2. Since the slide operation is unavailable, $d(v_1^0)+d(v_1^1)+d(v_1^2)+d(v_2^2) > 3$, so the vehicle is full. Note that both vehicles have been full, and after two tours no demand remains at level 1. This pattern continues. Tour t_i for $3 \leq i < p$ covers all demand at v_{i-1}^1, all remaining demand at v_{i-1}^2, and some demand at v_i^2. Inductively, since slide operation is unavailable and all previous vehicles have been full, this vehicle must also be full. After t_i, no demand remains below level i. Since all vehicles have been full, there is less than one unit of demand remaining for t_p (and no demand remains below level $p-1$), so t_p covers all demand from v_{p-1}^1 and all remaining demand from v_{p-1}^2. (See Figure 2b).

This cascading pattern results in i tours traversing e_i^0, one tour traversing e_i^1, two tours traversing e_i^2 for all i, and p tours traversing all edges in the path from the depot r to v_p^0. Note in particular that these values match the lower bounds given by the edge traffic, for e_i^0 and e_i^1. The excess cost, therefore, comes from doubling the traffic lower bound on the e_i^2 edges as well as extra traffic along the path to the depot.

5.2 Proofs of Lemmas 5 and 6

We restate the lemmas here for convenience. Recall that a branch is *resolved* if all of its demand is covered by a 4/3-approximate tour group. That is, a set of tours whose total cost is at most 4/3 times the reduction to the lower bound LB that results from removing the demand of the branch.

▶ **Lemma 5.** *Long p-chains can be resolved at the root.*

Proof. Let B be a long p-chain at the root (depot). Consider the cascade on B. As described in Section 5.1, the cost of the p tours in the cascade is

$$2\big(p \cdot l(e_p^0) + \sum_{i=1}^{p-1} i \cdot l(e_i^0) + 2l(e_i^2) + l(e_i^1)\big)$$

since the cost of the path to the depot is zero. Additionally the cascade covers all demand in B, so the reduction to LB after covering B is:

$$\sum_{e \in B} LB(e) = \sum_{e=(u,v) \in B} 2 \cdot l(e) \cdot \lceil d(T_v) \rceil = 2\big(p \cdot l(e_p^0) + \sum_{i=1}^{p-1} i \cdot l(e_i^0) + l(e_i^2) + l(e_i^1)\big)$$

Taking the ratio of cost to reduction in cost of LB gives our result:

$$\frac{2\big(p \cdot l(e_p^0) + \sum_{i=1}^{p-1} i \cdot l(e_i^0) + 2l(e_i^2) + l(e_i^1)\big)}{2\big(p \cdot l(e_p^0) + \sum_{i=1}^{p-1} i \cdot l(e_i^0) + l(e_i^2) + l(e_i^1)\big)} = 1 + \frac{\sum_{i=1}^{p-1} l(e_i^2)}{p \cdot l(e_p^0) + \sum_{i=1}^{p-1} i \cdot l(e_i^0) + l(e_i^2) + l(e_i^1)}$$

$$\leq 1 + \frac{\big(l(e_1^2)\big) + \big(\sum_{i=2}^{p-1} l(e_i^2)\big)}{\big(l(e_1^0) + l(e_1^1) + l(e_1^2)\big) + \big(\sum_{i=2}^{p-1} l(e_i^2) + l(e_i^1) + \sum_{j=i+1}^{p} l(e_j^0)\big)}$$

$$\leq 1 + \frac{\big(l(e_1^2)\big) + \big(\sum_{i=2}^{p-1} l(e_i^2)\big)}{\big(3 \cdot l(e_1^2)\big) + \big(\sum_{i=2}^{p-1} 3 \cdot l(e_i^2)\big)} \leq \frac{4}{3}$$

Where the second to last inequality comes from the fact that because B is a long p-chain, $l(e_i^2) < l(P[v_{i+1}^0, r]) = \sum_{j=i+1}^{p} l(e_j^0)$, and because $l(e_i^2) \leq l(e_i^1)$ by construction. ◀

▶ **Lemma 6.** *A long p-chain and long p'-chain can be resolved together if they are sibling branches.*

Proof. Let u be a vertex with child branches B a long p-chain and B' a long p'-chain. Consider the cascades on B and B'. The cost of these $p + p'$ tours is

$$2\Big[\big(p \cdot l(e_p^0) + \sum_{i=1}^{p-1} i \cdot l(e_i^0) + 2l(e_i^2) + l(e_i^1)\big) + \big(p' \cdot l(e'^0_{p'}) + \sum_{i=1}^{p'-1} i \cdot l(e'^0_i) + 2l(e'^2_i) + l(e'^1_i)\big) + (p + p')l(P[u,r])\Big]$$

Additionally the cascade covers all demand in B and B', so the reduction to LB is $\sum_{e \in B \cup B'} LB(e) = \sum_{e \in B \cup B'} 2 \cdot l(e) \cdot f(e)$, which is

$$2\Big[\big(p \cdot l(e_p^0) + \sum_{i=1}^{p-1} i \cdot l(e_i^0) + l(e_i^2) + l(e_i^1)\big) + \big(p' \cdot l(e'^0_{p'}) + \sum_{i=1}^{p'-1} i \cdot l(e'^0_i) + l(e'^2_i) + l(e'^1_i)\big) + (p + p' - 1)l(P[u,r])\Big]$$

Note that the $p + p' - 1$ factor in the reduction to the edges along $P[u, r]$ arises because by definition the total demand in a p-chain is between $p - 0.5$ and p, so covering all demand

in B and B' reduces the demand in T_u by at least $p + p' - 1$. Rearranging the terms, this reduction to the lower bound is greater than:

$$2\Big[\Big(\sum_{i=0}^{2} l(e_1^i) + \sum_{i=2}^{p-1} l(P[v_{i+1}^0, r]) + l(e_i^2) + l(e_i^1)\Big)$$

$$+\Big(\sum_{i=0}^{2} l(e'^i_1) + \sum_{i=2}^{p'-1} l(P[v'^0_{i+1}, r]) + l(e'^2_i) + l(e'^1_i)\Big) + 3l(P[u,r])\Big]$$

$$= 2(X + Y + X' + Y' + Z)$$

Where

$$X = \sum_{i=0}^{2} l(e_1^i)$$

$$Y = \sum_{i=2}^{p-1} l(P[v_{i+1}^0, r]) + l(e_i^2) + l(e_i^1)$$

$$X' = \sum_{i=0}^{2} l(e'^i_1)$$

$$Y' = \sum_{i=2}^{p'-1} l(P[v'^0_{i+1}, r]) + l(e'^2_i) + l(e'^1_i)$$

and

$$Z = 3l(P[u,r]).$$

The ratio of cost to reduction in cost of LB is therefore at most,

$$1 + \frac{\sum_{i=1}^{p-1} l(e_i^2) + \sum_{i=1}^{p'-1} l(e'^2_i) + l(P[u,r])}{X + Y + X' + Y' + Z} \leq \frac{4}{3}$$

Where the final inequality comes from noting that, by construction, $l(e_1^2) \leq X/3$ and $l(e'^2_1) \leq X'/3$, and by definition of a long p-chain, $l(e_i^2) \leq \min\{l(e_i^1), l(e_i^2), l(P[v_{i+1}^0, r])\}$ for all $2 \leq i < p$, so $\sum_{i=2}^{p-1} l(e_i^2) \leq Y/3$ and $\sum_{i=2}^{p-1} l(e'^2_i) \leq Y'/3$.

◀

References

1. Kemal Altinkemer and Bezalel Gavish. Heuristics for unequal weight delivery problems with a fixed error guarantee. *Operations Research Letters*, 6(4):149–158, 1987.
2. Tetsuo Asano, Naoki Katoh, and Kazuhiro Kawashima. A new approximation algorithm for the capacitated vehicle routing problem on a tree. *Journal of Combinatorial Optimization*, 5(2):213–231, 2001.
3. Tetsuo Asano, Naoki Katoh, Hisao Tamaki, and Takeshi Tokuyama. Covering points in the plane by k-tours: towards a polynomial time approximation scheme for general k. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 275–283. ACM, 1997.

4 Amariah Becker, Philip N Klein, and David Saulpic. Polynomial-time approximation schemes for k-center and bounded-capacity vehicle routing in metrics with bounded highway dimension. *arXiv preprint arXiv:1707.08270*, 2017.

5 Amariah Becker, Philip N Klein, and David Saulpic. A quasi-polynomial-time approximation scheme for vehicle routing on planar and bounded-genus graphs. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 87. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

6 Nicos Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, Carnegie-Mellon Univ Pittsburgh Pa Management Sciences Research Group, 1976.

7 Aparna Das and Claire Mathieu. A quasi-polynomial time approximation scheme for Euclidean capacitated vehicle routing. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 390–403. SIAM, 2010.

8 Bruce L Golden and Richard T Wong. Capacitated arc routing problems. *Networks*, 11(3):305–315, 1981.

9 Mark Haimovich and AHG Rinnooy Kan. Bounds and heuristics for capacitated routing problems. *Mathematics of operations Research*, 10(4):527–542, 1985.

10 Shin-ya Hamaguchi and Naoki Katoh. A capacitated vehicle routing problem on a tree. In *International Symposium on Algorithms and Computation*, pages 399–407. Springer, 1998.

11 Michael Khachay and Roman Dubinin. PTAS for the Euclidean capacitated vehicle routing problem in $\mathbb{R}^d$. In *International Conference on Discrete Optimization and Operations Research*, pages 193–205. Springer, 2016.

12 Martine Labbé, Gilbert Laporte, and Hélene Mercure. Capacitated vehicle routing on trees. *Operations Research*, 39(4):616–622, 1991.

13 Christos H Papadimitriou and Mihalis Yannakakis. The traveling salesman problem with distances one and two. *Mathematics of Operations Research*, 18(1):1–11, 1993.

A Showing That Safe Operations Are Safe

In this section we show that safe operations are actually safe. Recall that an operation that modifies the graph is *safe* if it does not decrease OPT or change the cost of the lower bound LB and it preserves feasibility.

The **condense** operation replaces a branch B in which every edge e has traffic $f(e) = 1$ with a single edge e_B of length $\sum_{e \in B} l(e)$ and traffic $f(e_B) = 1$, so the lower bound LB remains unchanged, since the branch contributes $2 \cdot \sum_{e \in B} l(e) \cdot 1 = 2 \cdot l(e_B) \cdot 1$ toward LB both before and after the operation. The operation is equivalent to requiring any tour that covers *any* client demand in B to traverse *all of* B, which clearly preserves feasibility and possibly *increases*, but cannot decrease, OPT.

Before the **unzip** operation, the contribution to LB of edges involved in the modification is $2 \cdot l(e) \cdot \left(\sum_{i=1}^{k} f((v, w_i)) \right) + \sum_{i=1}^{k} 2 \cdot l((v, w_i)) \cdot f((v, w_i))$ which equals $\sum_{i=1}^{k} 2 \cdot (l(e) + l((v, w_i))) \cdot f((v, w_i))$, the contribution to LB of involved edges *after* the operation. The length and traffic values (and thus also the contribution to LB) of all other edges remain unchanged. The operation is equivalent to requiring every tour that traverses c child edges of e to traverse e a total of $2c$ times. This redundancy may increase the value of OPT but cannot decrease it. Since any tour can be extended to cover e multiple times, the operation also preserves feasibility.

The **group** operation simply inserts an edge of length zero, resulting in an equivalent instance, and clearly has no affect on LB, OPT, or feasibility. The operation is merely for convenience.

The **unite** operation is equivalent to first inserting an edge of length zero (and traffic one) and then performing a condense operation, both of which are shown above to be safe. The composition of safe operations results in a safe operation.

The **slide** operation essentially *moves* an edge without changing the length or traffic values of any edge, which clearly preserves the value of LB. The operation is equivalent to requiring every tour that traverses edge e_2 to *also* traverse edge e_1, which may increase OPT, but cannot decrease OPT. Since the tree is connected, any tour can be extended to include e_1, so feasibility is preserved.

B Reforming Operations from Asano et al. [2]

Asano et al. introduce seven safe *Reforming Operations* [2]. In this section we compare these to the safe operations we use in this paper.

Reforming operation R_1 greedily takes a tour at full capacity to any vertex with demand greater than one. After exhaustively applying R_1, no vertex has demand greater than one. Reforming operation R_2 moves any demand at internal nodes to leaves, by adding edges of length zero when necessary.

In this paper, we use simplifying assumptions to accomplish the same result as exhaustive application of these two operations. That is, we assume that the input graph already has the property that leaves are the only vertices with demand, and that the demand of every leaf is at most one. Our algorithm does not also need do use these operations, because these properties of the demand are never *undone* by our algorithm.

Reforming operation R_3 addresses the specific case of a vertex u with a leaf vertex v of demand at most one as its only child, by contracting the edge (u, v), increasing $l((p(u), u))$ by $l((u, v))$, and setting $d(u) = d(v)$. Reforming operation R_4 similarly contracts an entire subtree with total demand at most one to a single leaf. In this paper, our **condense** operation generalizes both R_3 and R_4 as it operates on branches rather than subtrees. Note that R_3 also addresses the case where $1 < d(u) + d(v) \leq 2$, but this case never arises in our algorithm because of our simplifying assumptions.

Reforming operation R_5 merges leaves if their combined demand is at most one *and* if the parent vertex u of the leaves has no other child v such that the subtree rooted at v has total demand at least two. In this paper, our **unite** operation generalizes R_5.

Reforming operation R_6 addresses the case where a vertex v has exactly two children v_1 and v_2 that are leaves such that $1 < d(v_1) + d(v_2) < 2$. If v has parent u, R_6 removes v and adds edge (u, v_1) of weight $l((u, v)) + l((v, v_1))$ and edge (u, v_2) of weight $l((u, v)) + l((v, v_2))$. Our **unzip** operation generalizes R_6 to accommodate more than two children, non-leaf children, and larger demands.

Finally, reforming operation R_7 addresses the case where a vertex v has children v_1 and v_2 such that v_2 is a leaf and the subtree rooted at v_1 has total demand d_1 such that $1 < d_1 < 2$ *and* such that the subtree rooted at v has total demand d such that $1 < d < 2$. R_7 replaces edge (v, v_2) with edge (v_1, v_2) of length $l((v, v_2))$. Our **slide** operation generalizes R_7 to accommodate larger demand values.

C Example Showing Tightness

In this section we present the example of a tree instance that has cost at least $4/3 \cdot LB$ that was given by Asano et al. [2].

They define a tree T with root r, in which r has a single child q, and q has $2n+1$ children $v_1, v_2, ..., v_{2n+1}$ that are all leaves. Edge (r, q) as well as edges (q, v_i) for all $1 \leq i \leq 2n+1$ have length one. That is, T is a tree of height two in which all edges have unit lengths. Furthermore, the demand of each leaf v_i is $0.5 + \epsilon$, where $\epsilon < 1/(4n+2)$.

For this instance, the traffic of every edge (q, v_i) is one, and the traffic of edge (r, q) is $n+1$, so $LB = 2 \cdot 1 \cdot (n+1) + (2n+1) \cdot 2 \cdot 1 \cdot 1 = 2n + 2 + 4n + 2 = 6n + 4$.

An optimum solution for this instance consists of $2n+1$ tours $t_1, t_2, ..., t_{2n+1}$ in which tour t_i goes from r to v_i, covers all demand at v_i and then immediately returns to the depot.

The cost OPT of this optimum solution is $OPT = (2n+1) \cdot 4 = 8n + 4$.

The ratio of OPT to LB is $\frac{8n+4}{6n+4}$ which tends to $4/3$ as n goes to infinity [2].

Note that there are in fact many other optimum solutions for this instance. For example consider the following solution consisting of $n+1$ tours $t_1, t_2, ..., t_{n+1}$. For $i \in \{1, 2, ..., n\}$, t_i goes from r to v_{2i-1}, covers all demand at v_{2i-1}, then covers as much demand at v_{2i} as possible before returning to r. Tour t_{n+1} then covers all demand at v_{2n+1} and then all remaining demand at leaves v_{2i} for $i \in \{1, 2, ..., n\}$. The cost of this solution is $2 \cdot n \cdot 3 + 2 \cdot 1 \cdot (n+2) = 6n + 2n + 4 = 8n + 4$.

Low Rank Approximation in the Presence of Outliers

Aditya Bhaskara
School of Computing, University of Utah, Salt Lake City, UT, USA
http://www.cs.utah.edu/~bhaskara/
bhaskara@cs.utah.edu

Srivatsan Kumar
School of Computing, University of Utah, Salt Lake City, UT, USA
seezha@gmail.com

Abstract

We consider the problem of principal component analysis (PCA) in the presence of outliers. Given a matrix A ($d \times n$) and parameters k, m, the goal is to remove a set of at most m columns of A (outliers), so as to minimize the rank-k approximation error of the remaining matrix (inliers). While much of the work on this problem has focused on recovery of the rank-k subspace under assumptions on the inliers and outliers, we focus on the approximation problem. Our main result shows that sampling-based methods developed in the outlier-free case give non-trivial guarantees even in the presence of outliers. Using this insight, we develop a simple algorithm that has bi-criteria guarantees. Further, unlike similar formulations for clustering, we show that bi-criteria guarantees are unavoidable for the problem, under appropriate complexity assumptions.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis

Keywords and phrases Low rank approximation, PCA, Robustness to outliers

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.4

Funding The first author is partially supported by a Google Faculty Award.

1 Introduction

Low rank approximation is one of the most fundamental and well-studied questions in matrix analysis. It is widely used for dimension reduction, sketching, denoising, and more broadly to find "structure" in large matrices. Usually referred to as principal component analysis (PCA) or singular value decomposition (SVD), low rank approximations are a staple tool in the analysis of large matrix data.

We consider the problem of low rank approximation in the presence of *outlier* columns. Studying the effect of adversarial outliers on known algorithms as well as the problem complexity has become a significant theme in recent research. It is motivated by the practical importance of dealing with noise in data (both intrinsic as well as adversarial). While principal components are often robust to small corruptions of the matrix (which is why they are useful in denoising, e.g. [28]), this typically requires the noise to have a small spectral norm (see [31]). This is an unrealistic assumption in many settings, especially when entire columns can be (arbitrarily and possibly adversarially) noisy.

Dealing with noise has also been well studied in the statistics community (the books of [22, 23] present many of the classic results in the area). Robust estimators for computing parameters of distributions such as the mean, variance, etc. have been extensively studied. More recently, questions of this nature have received a lot of attention in theoretical computer

science, motivated by connections to learning distributions. The works of [12, 26], as well as subsequent works (see [30] and references therein) have obtained novel guarantees on recovery under noise, using semidefinite programming and other techniques.

Meanwhile, clustering problems have long been studied in the presence of outliers, from the point of view of approximation algorithms. Starting with the work of [6], and subsequent improvements (see [8, 18, 25]), we now have a good understanding of the approximability of clustering with outliers. Many of these works use linear programming relaxations that help identify the outliers. One of our motivations is to obtain a similar understanding for PCA.

PCA with outliers. We now define the formulation we study in this paper. It is motivated by the work on clustering algorithms discussed above. Informally, given a matrix A ($d \times n$), we wish to find the best rank k approximation to A, after throwing away m columns of our choice. This can be formally stated as the following optimization problem:

$$\text{minimize } \|A - L - N\|_F^2, \quad \text{subject to } \mathrm{rk}(L) \leq k, \text{ and } N \text{ having at most } m \text{ non-zero columns.}$$

This formulation is has also been studied in the signal processing literature [34, 5], where it is referred to as Robust PCA. (A different notion was given that name in [4].) The formulation also makes sense with different matrix norms, though in this paper, we focus mostly on the Frobenius norm. Our guarantees also work for entry-wise ℓ_p norm error, as we will see.

1.1 Background and related work

The statistics literature on robust estimation of parameters is extensive, and we refer to the book of [23] for a review. We now discuss some of the works most relevant to our results.

As mentioned above, the PCA with outliers problem has been well studied in the signal processing community. The work of [5] shows that by formulating the problem as an optimization problem with appropriate regularization terms, we can efficiently recover the optimal rank k subspace, under certain conditions on the input. Informally, these conditions say that the inliers must be "well spread", which in turn implie the uniqueness of the target subspace (this is similar in spirit to works on matrix completion). The worst-case problem was studied in [32], under a "coverage" variant of the objective (in which the goal is to find a rank k subspace that covers as large a fraction of the inlier mass as possible). The results here do not imply a multiplicative approximation to our formulation, and are thus incomparable.

We also note that while the problem definition naturally suggests $m \ll n$, the problem makes sense when we have a large fraction (approaching 1) of outliers. This version was studied in the work of [20]. They formulate the problem as follows: given a set of points in $\mathbb{R}^n$ with the condition that an α fraction lie in a d-dimensional subspace, the goal is to find the subspace. Informally, under the condition that the "outliers" (the points not in the subspace) are in general position, they develop a simple random sampling algorithm, for the case $\alpha > d/n$. They also show hardness results for this problem. Indeed, we note that the hardness results we present in Section 5 are consequences of their approach.[1]

The main issue with $m \approx n$ is that in the absence of any strong assumptions (along the lines of the above works), the outliers could themselves have an approximate rank-k structure, thus making the problem ill-posed. For problems such as mean estimation, [7] recently showed that in such cases, something very elegant is possible: we can come up with

[1] We thank Amit Deshpande and Ravishankar Krishnaswamy for suggesting this.

a small set of *candidate* solutions, one of which will be a good solution to the inliers. This framework was originally introduced in [1], and together with some very interesting new ideas, it has been used to obtain results for classic problems such as learning mixtures of Gaussians under separation [13, 24, 21].

Finally, as we mentioned above, outlier-robust approximation algorithms are quite well studied for different variants of clustering (see [6, 8, 18, 25] and references therein). For many variants, while the initial algorithms were bi-criteria approximations similar to ours, it turns out that one can actually obtain constant factor approximation algorithms without violating either the bound on the number of centers, or the number of outliers. This is in contrast to what turns out to be possible for PCA, as we will see.

Robust algorithmic methods. The high level goal in the works above (as well as ours) is the development of algorithmic techniques that work in the presence of outliers. To this end, linear and semidefinite relaxations have been powerful in identifying the inliers/outliers, and have emerged as a powerful technique. This is seen in the works of [12, 7], the works on clustering mentioned above, and also the extensive literature on semi-random models (see [14, 27] and references therein). Recently, [30] studied the abstract question of when an estimator can be computed robustly, and defined a condition they call *resilience*. This is a property of the inliers that allows estimation (in principle) in the presence of a small fraction of arbitrary outliers. In this context, our work shows that random sampling based algorithms could be powerful in finding structure in the presence of noise, albeit with weaker (bi-criteria) guarantees. This is the idea behind heuristics such as RANSAC, which we will now discuss.

Heuristics. There have been several heuristics developed specifically for the robust PCA problem, as well as more generally for estimation on noisy data. One example is the class of "Lloyd-style" heuristics, where the idea is to fit a rank-k subspace to the full data set, remove a small number of points that are "far away" from the space, and recurse (see [18, 33]). Another heuristic the famous RANSAC (random sampling and consensus) paradigm [15]. The idea here is that if we randomly sample a subset of the points and fit a k-subspace, then different samples yield spaces that align "along the inliers" but have differing components along the outliers. Assuming the outliers do not have "structure", we can expect that an averaging (consensus) step helps zero in on the inliers. One issue with the above is that the outliers could have an approximate rank k structure. Intuitively, this is one of the reasons for which we only obtain bi-criteria guarantees.

1.2 Our results

In the remainder of the paper, we will write $A = B + N$, where B consists of the inliers (and zeros in the outlier columns), and N contains the outliers (and zeros in the inlier columns). Thus, in this notation, the problem can be reformulated as that of finding such a decomposition $B + N$ where N has at most m non-zero columns, with the goal of minimizing $\|B - B_k\|_F^2$, where B_k is the best rank-k approximation of B. Also, we will throughout think of ϵ, δ as parameters in the range $(0, 1]$.

We present two (incomparable) algorithmic results. The first is a simple algorithm based on iteratively computing a subspace that captures more and more "mass" of the matrix, while throwing away a small number of outliers. The second algorithm, which is our main contribution, is inspired by *adaptive sampling*, an idea that has been successful in obtaining

coresets and bi-criteria approximations for problems such as clustering and PCA [11, 9]. To describe the first result, we define the following "rank-k condition number":

$$\Lambda_k := \frac{\|A\|_F^2}{\|B - B_k\|_F^2}.$$

▶ **Theorem 1.** *There is an efficient algorithm that takes as input a matrix A as above, parameters k, m, ϵ, and outputs a decomposition $A = B' + N'$ along with a subspace V, such that the following properties hold: (a) N' has at most $m \log(\Lambda_k/\epsilon)$ columns, (b) the space V satisfies the property*

$$\|\Pi_V^{\perp} B'\|_F^2 \leq (1+\epsilon)\|B - B_k\|_F^2,$$

and (c) the dimension of V is at most $k \log(\Lambda_k/\epsilon)$.

The algorithm above violates the bounds on the number of outliers and the dimension of the space by a factor $\log(\Lambda_k/\epsilon)$. Thus it is interesting in the regimes where m is small compared to the number of columns of A, and the quantity Λ_k is "not too large". For instance, in some practical settings, one might be interested in capturing say 99% of the mass of a matrix using a small subspace, while excluding a small number of outliers. In such a case, the Λ_k is a constant, and all the additional factors are small.

Our second (and main) result has a much better dependence on the parameters. It has as an additional input a parameter δ, which controls the number of outliers the algorithm outputs.

▶ **Theorem 2.** *There is an efficient algorithm that takes as input a matrix A as above, parameters k, m, ϵ, δ, and outputs a decomposition $A = B' + N'$ along with a subspace V, such that the following properties hold: (a) N' has at most $(1+\delta)m$ columns, (b) the space V satisfies the property $\|\Pi_V^{\perp} B'\|_F^2 \leq (1+\epsilon)\|B - B_k\|_F^2$, and (c) V is the span of a subset of the columns of A, of size at most*

$$O\left(\frac{k}{\epsilon^6}\left(\log(n/m) + \frac{2}{\delta}\right)\right).$$

Thus, the theorem in fact gives a "column based" approximation to the space V. Also, if we think of ϵ as a constant, the algorithm outputs $O(k \log(n/m) + k/\delta)$ columns. Note that in many cases of interest, we may have m being a small constant factor of n. In such cases, the algorithm roughly outputs only a $O(k/\delta)$ dimensional space, while obtaining a $(1+\epsilon)$ approximation to the objective and violating the number of constraints by a factor $(1+\delta)$.

Dependence on ϵ. The drawback in Theorem 2 is the dependence on ϵ. Indeed, the first algorithm, while lossy in many other aspects, has a really good dependence on ϵ (which is what makes the algorithms incomparable). However, we note that even in the noise-free case, obtaining a column-based $(1+\epsilon)$ approximation to the best rank-k approximation requires k/ϵ columns (see [19]). Thus we cannot hope to get rid of $1/\epsilon$ entirely.

Technique and extensions. It turns out that the key to Theorem 2 is a modification of a remarkable lemma from [9], on finding low-rank approximations under entry-wise ℓ_p error by using iterative *uniform* sampling. While the modification (see Lemma 6 and the notes following it) is needed for our main result, it turns out that if we use the original lemma of [9] in our framework, we obtain the following as a corollary. Note that the approximation ratio is now $O(k)$ as opposed to $(1+\epsilon)$ for the case of Frobenius norm. (A dependence on k turns out to be unavoidable for column-based approximations for ℓ_p error even without noise [9].)

▶ **Theorem 3.** *There is an efficient algorithm that takes as input a matrix A as above, parameters k, m, δ, and outputs a decomposition $A = B' + N'$ along with a subspace V, such that the following properties hold: (a) N' has at most $(1+\delta)m$ columns, (b) the space V is spanned by $O(k(\log(n/m) + 1/\delta))$ columns of A, and (c) the error satisfies*

$$err(p, B', V) \leq 100(k+1) \cdot \min_{X \in \mathbb{R}^{d \times k}, Y \in \mathbb{R}^{k \times n}} \|B - XY\|_p^p,$$

where $err(p, B', V)$ denotes the minimum ℓ_p^p reconstruction error of the columns of B' using V.

Limits of approximation. It is natural to ask if there needs to be a trade-off between the dimension of the output space and the slack parameter δ. Furthermore, we can even ask if we can avoid having a slack altogether.

By a reduction along the lines of the result of [20], the following result is quite easy to show.

▶ **Theorem 4** (Informal version of Theorem 12)**.** *Under the small set expansion conjecture with suitable parameters, for any constant $C > 0$, there is no polynomial time algorithm that can obtain a multiplicative factor approximation to the objective, while returning a Ck dimensional subspace, and excluding at most $(1+\delta)m$ outliers, for small enough constant δ.*

This rules out the possibility of finding a Ck-dimensional subspace for arbitrarily small δ. A very similar reduction, but from the smallest r-edge subgraph problem (see Section 5.2) implies that if we wish to have $\delta = 0$, then the dimension of the subspace output must be at least $k \cdot n^{\Omega(1)}$. See Corollary 13 for the formal statement. We remark once more that these hardness results indicate that "pure" approximations (as can be obtained for clustering) are impossible for PCA.

Open problems, directions. While bi-criteria guarantees are unavoidable in the worst case, it is interesting to see if simple iterative algorithms like the ones we proposed can be shown to *recover* the k-PCA subspace of the inliers under appropriate assumptions. A starting point would be assumptions similar to the ones of Donoho et al. Next, the dependence on δ, ϵ in our sampling based algorithm are possibly sub-optimal. It would be interesting to see if more sophisticated algorithms can give better guarantees. More broadly, it would be interesting to show more guarantees for heuristics such as RANSAC and algorithms inspired by them for other problems involving outliers.

2 Notation and preliminaries

We start with some basic matrix notation we use. Let A be a $d \times n$ matrix. Throughout the paper, we write A_k to refer to the best rank-k approximation of A (thus it is also a $d \times n$ matrix). For a subset T of the column indices ($T \subseteq [n]$), we denote by $A_{(T)}$ the $d \times |T|$ sub-matrix of A formed by the columns indexed by T.

Next, given an integer k, we denote by $err_k(A)$ the error in the best rank-k approximation of A. Specifically, $err_k(A) = \|A - A_k\|_F^2$. Also, for a set of vectors W, their linear span is denoted span(W). Finally, the projection matrix orthogonal to the space orthogonal to span(W) will be denoted by $\Pi_W^{\perp}$. So also, for a *subspace* W, we abuse notation slightly and denote by $\Pi_W^{\perp}$ the projection matrix to the space orthogonal to W.

Algorithm 1 Iterative SVD.

Input: Matrix $A \in \mathbb{R}^{d \times n}$, guess ξ for the optimum error, parameter m (bound on # outliers), and accuracy parameter ϵ.

Output: A subspace V, and a set S of inliers.

Initialize $V_0 = \emptyset$, $S_0 = \text{cols}(A)$, and $j = 0$.
while total squared projection of S_j onto the space orthogonal to V_j is $\geq (1+\epsilon)\xi$ **do**
  Let $u_1, u_2, \ldots, u_N$ be the projection of the columns in S_j orthogonal to V_j.
  Define $\mu_j := \sum_i \|u_i\|^2$.
  Let T_j be the m largest (by length) vectors among $\{u_i\}_{i=1}^N$.
  if $(\sum_{u \in T_j} \|u\|^2 \geq \frac{1}{2}(\mu_j - \xi)$) **then**
    $S_{j+1} := S_j \setminus T_j$, and $V_{j+1} = V_j$.
  else
    Let V_{j+1} be the rank-$((j+1)k)$ SVD for S_j.
    Set $S_{j+1} = S_j$.
  end if
  $j \leftarrow j + 1$
end while
return V_j, S_j

Guessing the optimum. In all our algorithms, we assume that we have a guess for the optimum error (error in the low-rank approximation of B), up to a multiplicative factor of $(1+\epsilon)$. A fairly straightforward argument shows that we can always come up with a polynomial number (in the input complexity) of guesses, one of which is accurate. This is shown in Appendix A.

3 Iterative SVD

We now present our first algorithm. It involves repeatedly computing the best low rank approximation, while potentially throwing away some points as outliers. This will establish Theorem 1. Let us start with an informal description of the algorithm.

Algorithm outline. At each step j, we have a subset S_j of the initial column vectors, and a subspace V_j. Let $N = |S_j|$ and let $u_1, u_2, \ldots, u_N$ denote the projections of the columns in S_j, orthogonal to the space V_j. The aim is to either (a) find a new subspace V_{j+1} that captures a significantly larger fraction of the total mass than S_j, or (b) remove a set of at most m columns from S_j and mark them as outliers. In either case, we show that the total "uncaptured" mass remaining drops by a constant factor. This allows us to bound the number of iterations, thus giving the guarantees of Theorem 1.

The algorithm is formally stated below (Algorithm 1). As discussed in Section A, it assumes that we have a guess ξ for the optimum error.

▶ **Lemma 5.** *In every iteration of the algorithm, the total mass of the inliers reduces significantly. More precisely, we have* $\mu_{j+1} \leq (\mu_j + \xi)/2$.

Proof. We consider both the cases in the algorithm.

Case 1. $\sum_{u \in T_j} \|u\|^2 \geq \frac{1}{2}(\mu_j - \xi)$.

In this case, we remove T_j from the set of inliers, and thus

$$\mu_{j+1} = \mu_j - \sum_{u \in T_j} \|u\|^2 \leq \frac{\mu_j + \xi}{2}.$$

Case 2. $\sum_{u \in T_j} \|u\|^2 < \frac{1}{2}(\mu_j - \xi)$.

Let us denote the set of inlier columns by $\mathcal{I}$. Let us argue about the error in the best rank-$(j+1)k$ approximation of S_j. To do so, we consider the space $V' = V_j + V^*$, where V^* is the rank-k SVD space of $\mathcal{I}$ (this is the optimal subspace that we are after). Now, let us consider the projection of the columns of S_j orthogonal to V'. For the columns $S_j \cap \mathcal{I}$, the total error has to be $\leq \xi$, because by assumption, projecting $\mathcal{I}$ orthogonal to V^* has this error. Next, the projection of $S_j \setminus \mathcal{I}$ orthogonal to V' must be smaller than (or equal to) the projections orthogonal to V_j. As $|S_j \setminus \mathcal{I}| \leq m$ (there are at most m outliers), their projections orthogonal to V_j can be bounded by $\sum_{u \in T_j} \|u\|^2$ (as T_j contained the m largest vectors by length). This allows us to bound the total error by

$$\xi + \frac{\mu_j - \xi}{2} = \frac{\mu_j + \xi}{2}.$$

The best $(j+1)k$ dimensional subspace will result in an error only better than the above, and thus the lemma follows.[2] ◀

The lemma above immediately implies that (assuming that the guess of ξ is an upper bound on the optimum), the algorithm runs for at most $\log(\|A\|_F^2/\epsilon\xi)$ iterations. This is because the gap between μ_j and ξ drops by a factor at least 2 in each iteration. Thus, by searching over all the possible ξ and by the comment below, Theorem 1 follows.

Validating the guess of ξ. We note that the above algorithm works whenever ξ is an upper bound on the optimum. If it is lower than the optimum, then in one of the iterations, we may not see a drop in μ_{j+1} as guaranteed by Lemma 5. Thus, we can test for this as the algorithm proceeds and output FAIL if we do not see a drop.

4 Iterative uniform sampling

We next present our main algorithm for the PCA with outliers problem. We will start with a sampling lemma that is at the heart of the algorithm. This lemma applies to the case when there are no outlier columns. As mentioned earlier, the lemma is a variant of a lemma from [9], which applies to ℓ_p norms and has additional factors of k. In Section 4.2, we show how the lemma can be used even in the presence of outliers, thus establishing Theorem 2.

4.1 Sampling without outliers

The first lemma is simply about low rank approximation via columns (without any outliers).

[2] We note that a more "natural" algorithm is to add the top-k SVD space of the u_i vectors to the current space in step 9. This is closer to adaptive sampling paradigm (see [11]), but it runs into the issue that it is not clear the projection of the u_i corresponding to inliers reduces to ξ. This stems from the fact that for two spaces V, W, $\Pi_V u$ could be smaller in length than $\Pi_V(\Pi_W u)$.

▶ **Lemma 6.** *Let $A \in \mathbb{R}^{d\times n}$, and let $\epsilon \in (0,1]$ be any parameter. Let S be a uniformly random subset of $[n]$ of size s, where $s \geq 4k/\epsilon^2$. Then, with probability at least $\epsilon^2/8$, there exist a set of $\epsilon^2 n/8$ columns of A, whose projection orthogonal to the column span of $A_{(S)}$ is upper bounded by $(1+\epsilon)\|A - A_k\|_F^2/n$.*

Note. The lemma is quite surprising: for say $\epsilon = 1$, it says that a *uniformly random* subset of $4k$ columns covers a constant fraction of the columns up to a constant times the k-SVD error. Such a guarantee is not even clear for norm-based sampling (see [16]). So, while uniform sampling does not necessarily give a low *total error* in expectation, it ends up giving small error for a constant fraction of columns.

We use the same rough outline as the proof of [9]. There are, however, two main differences. First, as we need a column-based $(1+\epsilon)$ guarantee on rank-k approximation, we appeal to the results of [19], instead of a weaker $O(k)$ bound used in [9] for the ℓ_p norm. Second (and more significant), their proof uses a simple union bound over failure probabilities. This does not suffice for a $(1+\epsilon)$ approximation, as two of the events are low probability (roughly ϵ). We observe that these events are effectively independent, and so we obtain a constant probability of success. .

Proof. Let us denote $s = 4k/\epsilon^2$. Let T be a random subset of $[n]$ of size $s+1$. We think of sampling S as first sampling T and then removing a random element $u \in T$. For convenience, let us write

$$\theta = \frac{\|A - A_k\|_F^2}{n}.$$

First, let us call a subset T *good* if $A_{(T)}$ has a small error rank-k approximation. Concretely, T is said to be good if $err_k(A_{(T)}) \leq (1+\epsilon)\cdot|T|\theta$. Now, if Π_k is the projection matrix onto the space orthogonal to the best rank k approximation for the full matrix A, we have $\mathbb{E}[\sum_{u\in T}\|\Pi_k u\|^2] = |T|\theta$ (where the expectation is over the choice of T). Thus, by Markov's inequality, we have

$$\Pr\left[\sum_{u\in T}\|\Pi_k u\|^2 > (1+\epsilon)|T|\theta\right] \leq \frac{1}{1+\epsilon} \leq 1 - \frac{\epsilon}{2}. \tag{1}$$

Thus, as the best rank-k approximation for $A_{(T)}$ can only have a smaller error, we have that T is good with probability at least $\epsilon/2$. Next, we show the following claim.

▶ **Claim 1.** *For any T of size $\geq 4k/\epsilon^2$, if we form S by randomly removing a $u \in T$, then with probability at least $\epsilon/2$ (over the choice of u), we have*

$$\|\Pi_S^{\perp} u\|^2 \leq (1+\epsilon)\cdot\frac{err_k(A_{(T)})}{|T|}. \tag{2}$$

To show this claim, we first appeal to the existence of good column-based low-rank approximations. Specifically, Guruswami and Sinop showed the following.

▶ **Theorem 7** (Guruswami, Sinop [19])**.** *For any matrix B and parameter k, there exist a subset W of at most k/ϵ columns of B with the property that*

$$\|\Pi_W^{\perp} B\|_F^2 \leq (1+\epsilon)\cdot err_k(B).$$

We apply the result to the matrix $A_{(T)}$, and let W denote the set of columns guaranteed by the theorem. Now, the probability that a random column u of $A_{(T)}$ belongs to W is at most $|W|/|T|$, which is at most $\epsilon/2$, by our choice of s. Thus, the probability that $S \supset W$ is at least $1 - \epsilon/4$.

Next, we also have that for a random column u of $A_{(T)}$, the expected value

$$\mathbb{E}[\|\Pi_W^{\perp} u\|^2] = \frac{\|\Pi_W^{\perp} A_{(T)}\|_F^2}{|T|} \leq (1+\epsilon)\ \frac{err_k(A_{(T)})}{|T|}.$$

Once again, by Markov's inequality, the probability that $\|\Pi_W^{\perp} u\|^2$ is bounded by $(1+\epsilon)$ times the RHS is at least $\epsilon/2$. Thus by a union bound, with probability at least $\epsilon/4$, we have this condition, as well as the event $S \supset W$. In this case, we clearly have $\|\Pi_S^{\perp} u\| \leq \|\Pi_W^{\perp} u\|$, and this completes the proof of Claim 1.

Now, consider the bipartite graph in which the left side consists of all $(s+1)$-tuples of $[n]$ and the right side consists of all s-tuples of $[n]$. We place an edge between T and S if (a) T is good, and (b) $u = T \setminus S$ satisfies Eq. (2). The claim above, together with Eq. (1) (which lower bounds the probability of T being good), imply that the total number of edges is at least

$$\frac{\epsilon}{2}\binom{n}{s+1}\frac{\epsilon}{4}(s+1) = \frac{\epsilon^2}{8}(n-s)\binom{n}{s}.$$

Note that $n - s$ is the maximum number of edges that could be incident to a vertex on the right. Thus, we conclude that at least an $\epsilon^2/8$ fraction of the vertices on the right have degree at least $\epsilon^2(n-s)/8$. Since an edge implies that (a) T is good and (b) Eq. (2) holds, the conclusion of the lemma follows. ◀

4.2 Incorporating outliers

Next, suppose the set of columns contains (at most) m outliers. We then consider Algorithm 2. The analysis will use the value of the average error per column in the optimum solution, namely $\theta := \|B - B_k\|_F^2/(n-m)$.

Define $n' = n - m$. Let $z_1, z_2, \ldots, z_{n'}$ be the rank k approximation errors of the columns B_i of B. (So $\sum_i z_i = n'\theta$.) We will assume (without loss of generality) that z_i are in increasing order.

Proof outline. Consider the first iteration of the algorithm. We claim that $\|\Pi_{R^*} A^*\|_F^2 \leq (1+\epsilon)(z_1 + z_2 + \cdots + z_{\epsilon n'})/(\epsilon n')$ with probability at least $1 - \frac{1}{n^4}$. This is because with high probability, one of the candidates for R would have chosen $4k/\epsilon^2$ columns *among* $B_1, \ldots, B_{\epsilon n'}$, and then we can apply Lemma 6. Thus, the error for the first $n'\epsilon^3$ columns covered is at most the average error for the first $\epsilon n'$ columns. Subsequently, we will be able to bound the error in each step in terms of the smallest ϵ fraction of the z_i's that remain.

▶ **Lemma 8.** *Consider the procedure* SELECT$(A', m, k, \epsilon, \delta)$*, and suppose that* $N := \#cols(A')$ *satisfies* $N \geq \max\{n_0, (1+\delta)m\}$*, where* n_0 *is defined in step 3 of the algorithm. Let* $B' = A' \cap B$*, i.e., the set of inliers in* A'*, and let* $y_1, y_2, \ldots, y_{|B'|}$ *denote the values* z_i *for* $i \in B'$*. Assume w.l.o.g. that* y_i *are in increasing order. Then with probability* $\geq 1 - \frac{1}{n^4}$*, the sets* R^*, A^* *chosen by the procedure satisfy*

$$\text{for all } u \in A^*,\ \|\Pi_{R^*}^{\perp} u\|_F^2 \leq \frac{\sum_{i=1}^{\epsilon|B'|} y_i}{\epsilon|B'|}. \tag{3}$$

Algorithm 2 Iterative sampling with outliers.

Input: Matrix $A \in \mathbb{R}^{d\times n}$, parameters m, k, δ, ϵ, guess θ for optimum error per column.
Output: A set of outliers $\mathcal{O}$ and a set of columns V of A.

1: **procedure** SELECT($A, m, k, \epsilon, \delta$)
2: Initialize $\mathcal{O} = V = \emptyset$. Define $N = \#\text{cols}(A)$.
3: Define $n_0 = \frac{\alpha}{\alpha-1} \cdot \frac{8k}{\epsilon^3}$, where $\alpha = N/m$.
4: **if** $N < n_0$ **then return** add all the columns of A to V and return $(\mathcal{O}, V)$
5: **else if** $N \leq (1+\delta)m$ **then**
6: add all the columns of A to $\mathcal{O}$ and return $(\mathcal{O}, V)$
7: **else**
8: **for** $16 \log n/\epsilon^2$ iterations **do**
9: Let $R \leftarrow$ a uniformly random sample of n_0 columns of A
10: Let $\hat{A}$ be the set of $\epsilon^3(N-m)$ columns of A that have the least projection to $\Pi_R^{\perp}$
11: Let $X = \|\Pi_R^{\perp}\hat{A}\|_F^2$
12: If X is smaller than the least such quantity so far, set $A^* = \hat{A}$, $R^* = R$
13: **end for**
14: Mark the columns A^* as *covered*
15: Let $(\mathcal{O}', V')$ be the output of the recursive call SELECT($A \setminus A^*, m, k, \epsilon, \delta, \theta$)
16: **return** $(\mathcal{O}', R^* \cup V')$
17: **end if**
18: **end procedure**

Proof. For each iteration of the loop (8-13) of the algorithm, we show that the probability of the chosen $R, \hat{A}$ satisfying the bound in Eq. (3) is at least $\epsilon^2/2$. The conclusion of the lemma then follows immediately.

First, note that for $\alpha = N/m$, we have $|B'|/|A'| \geq (\alpha-1)/\alpha$. Let us also denote by Q the set of $\epsilon|B'|$ columns of B' that have the smallest z_i values. Now, the expected size of $R \cap Q$ is

$$\frac{|R|}{|A'|} \cdot \epsilon|B'| \geq \frac{8k}{\epsilon^2}.$$

Thus the probability that this is $\geq 4k/\epsilon^2$ is at least $1/2$. (Formally, this follows from Hoeffding's inequality, which also applies to sums of random variables without replacement.)

Conditioned on $|R \cap Q| \geq 4k/\epsilon^2$, we can apply Lemma 6 to conclude that with probability $\geq \epsilon^2$, at least an ϵ^2 fraction of the columns in Q, the projection orthogonal to span(R) is bounded by $\sum_{i\in Q} z_i/|Q|$. Note that by definition, this is precisely the RHS of Eq (3). Thus the probability that the chosen $R, \hat{A}$ satisfy (3) is at least $\epsilon^2/2$, and this completes the proof of the lemma. ◀

▶ **Lemma 9.** *When the algorithm terminates, the total error incurred is bounded by* $\frac{1+\epsilon}{1-\epsilon}n'\theta$, *with high probability. Further, the depth of the recursion is upper bounded by* $\frac{\log(n/(\delta m))}{\epsilon^3}$. *The total number of columns chosen is at most*

$$\frac{16k}{\epsilon^6}\left(\log(n/m) + \frac{2}{\delta}\right).$$

Proof. Using Lemma 8, we can analyze the overall error as follows. Since the success probability in each iteration is $1 - \frac{1}{n^4}$ we assume henceforth that the conclusion of Lemma 8

holds for all the recursive calls. Let us divide the indices $[n']$ (from left to right) into groups of size $\epsilon n', \epsilon(1-\epsilon)n', \epsilon(1-\epsilon)^2 n', \ldots$. Let us call the groups $G_1, G_2, \ldots$, and let E_i denote $\sum_{j \in G_i} z_j$. Now, by the lemma, until the algorithm marks $\epsilon n'$ columns as covered, we have that the average error in the marked columns is at most $(1+\epsilon)E_2/|G_2|$, w.h.p. (Note that the set of columns marked by the algorithm can be quite different from G_1; but this will only help the argument, which only requires that an ϵ fraction of the uncovered columns has average error $\leq E_2/|G_2|$.) Likewise, until the next $\epsilon(1-\epsilon)n'$ columns are marked covered, the average error is $\leq (1+\epsilon)E_3/|G_3|$. This continues until the number of unmarked columns falls below k/ϵ^3, at which point the error will be zero, as all the columns will be picked.

Thus, we cover the first $|G_1|$ columns with average error $\leq (1+\epsilon)E_2/|G_2|$, the next $|G_2|$ columns with average error $\leq (1+\epsilon)E_3/|G_3|$, and so on. And when $|G_i|$ gets to k/ϵ^3 (or δm), the error is zero and the procedure stops. Since $|G_i|/|G_{i+1}| = 1/(1-\epsilon)$, we can bound the total error by

$$(1+\epsilon)\frac{E_2 + E_3 + \ldots}{1-\epsilon} \leq \frac{(1+\epsilon)n'\theta}{1-\epsilon}.$$

This completes the proof of the error bound.

For the depth of the recursion, note that we always mark precisely $\epsilon^3(N-m)$ columns as marked. The procedure terminates when $N-m$ is $\max\{n_0, (1+\delta)m\}$, and this gives the desired bound.

To bound the number of columns chosen, we need to analyze the sum of n_0 over the recursive calls. We note that as long as $\alpha \geq 2$, we have $n_0 \leq 16k/\epsilon^3$. By the argument above, the number of recursive steps needed to reach $\alpha = 2$ is at most $(1/\epsilon^3) \cdot \log(n/m)$. This bounds the number of columns chosen until this step by $(16k/\epsilon^6)\log(n/m)$. Next, as $(\alpha - 1)$ drops from 2^{-j} to 2^{-j-1}, we end up with $n_0 \leq 2^{j+1} \cdot 8k/\epsilon^3$ columns in each call to Select. The number of recursive calls necessary for this drop in α is at most $1/\epsilon^3$. Thus, summing over j from 0 through $\log(1/\delta)$, we have that the number of columns chosen is at most

$$\sum_{j=0}^{\log(1/\delta)} 2^{j+1}\frac{8k}{\epsilon^6} \leq \frac{32k}{\epsilon^6\delta}.$$

This completes the proof of the lemma. ◀

Theorem 2 now follows easily.

Proof of Theorem 2. The desired bound on the number of columns, as well as the bound on the approximation ratio and the number of outliers all follow from Lemma 9. ◀

4.3 Entry-wise ℓ_p error

We notice that the algorithm above can easily be adapted to the case in which we care about the entry wise ℓ_p error. Again, we have a sampling lemma in the setting without outliers. As mentioned earlier, the following lemma was shown in [9].

▶ **Lemma 10** (Lemma 6 of [9]). *Let $A \in \mathbb{R}^{d\times n}$, and let $A_{k,p}$ be a minimizer of $\|A - X\|_p$ over rank-k matrices X. Let S be a uniformly random subset of $[n]$ of size $2k$. Then, w.p. at least $1/10$, there exists a set of $n/10$ columns of A whose optimum ℓ_p^p reconstruction error using the columns of $A_{(S)}$ is at most $100(k+1)\|A - A_{k,p}\|_p^p/n$.*

We can now apply the reasoning from earlier, along with a modified algorithm, in which we treat a column as "covered" if the ℓ_p^p reconstruction error using the columns chosen is at most $100(k+1)/n$ times the guess for the optimum. One component we need here is to be able to compute the optimum ℓ_p^p reconstruction error. This can be done via a convex program for any $p \geq 1$. We refer to [9] for the details. By following the proof earlier, we obtain Theorem 3. We omit the details.

The number of columns is now $O\left(k(\log(n/m) + 1/\delta)\right)$.

5 Limits of approximation

We make some simple observations about the computational complexity of the problem of PCA with outliers. First, we consider the consequences of a reduction due to Hardt and Moitra [20] to our setting. This yields a strong impossibility result assuming the small set expansion (SSE) conjecture (see [29]). Then, we show that if we do not allow a slack in the number of outliers, then we cannot even hope to find a "reasonably small" dimensional subspace with an error within any multiplicative factor.

Notes on our reductions. Before proceeding, we note the following

1. In the reductions, the number of outliers is very close to the total number of columns. While this is intuitively not the regime of "practical interest", it is easy to see that by padding $O(n)$ copies of a vector orthogonal to all the ones produced in the reduction, we (a) obtain the setting in which the number of outliers is a small constant fraction of the total number of columns, and (b) have the same lower bounds (because a change of ± 1 to the target dimension does not matter in our proofs). Intuitively, these are cases in which one of the components is easy to find, and it becomes trickier to find the others. These are precisely the type of pathologies avoided by the "well spread" assumptions in [5, 34].
2. Next, we note that both the reductions are in the case when the optimum error is zero. Thus, our lower bounds imply that we cannot obtain *any* multiplicative approximations under the corresponding assumptions.

5.1 Reduction from SSE

Hardt and Moitra [20] give a reduction from small set expansion (SSE) to *robust subspace recovery*, a problem closely related to PCA with outliers. Let us start by recalling the SSE(δ, ϵ) problem. We are given a Δ-regular graph $G = (V, E)$ on n vertices, and the goal is to distinguish between the following two cases

YES: there exists a set S of size δn with $\Phi(S) \leq 1/2$.[3]
NO: every set of size $\leq \delta n$ has expansion $\geq 1 - \epsilon$.

We note that typically the SSE problem is stated with the YES case having $\Phi(S) \leq \epsilon$. The version above is clearly at least as hard. The "SSE conjecture" [29] states that for any $\epsilon > 0$, there is a small enough $\delta > 0$ such that SSE(δ, ϵ) is hard (for polynomial time algorithms).

Now, the reduction of [20] constructs a collection of vectors in $\mathbb{R}^{|V|}$, one for each edge. The edge $\{i, j\}$ corresponds to the vector $\mathbf{e}_i + \mathbf{e}_j$, where $\mathbf{e}_i$ denotes the unit vector of the standard basis. For an edge f, we denote this by $v(f)$. The main lemma behind their reduction is the following:

[3] As is standard, $\Phi(S)$ denotes the conductance of S, namely the quantity $\frac{E(S,\overline{S})}{\Delta \cdot \min\{|S|, n-|S|\}}$.

▶ **Lemma 11** (Hardt, Moitra [20]). *Let E' be a subset of edges, and let n' be the total number of vertices that these edges are incident to. Then we have*

$$n'/2 \leq \dim(span(\{v(f) : f \in E'\})) \leq n'.$$

The proof is simple, and proceeds as follows. First, the upper bound is trivial, as all the vectors are spanned by the vectors in the standard basis corresponding to the vertices incident to E'. The lower bound follows from arguing that for any spanning tree T', the vectors $v(f)$ corresponding to the edges of T' are linearly independent. This can be shown by way of contradiction. Suppose there exist α_f such that $\sum_{f \in T'} \alpha_f v(f) = 0$. The edges corresponding to the non-zero coefficients form a forest. Now consider any *leaf* j, i.e., a vertex that is incident to precisely one edge in the forest (which has to exist). Then $\langle \mathbf{e}_j, \sum_f \alpha_f v(f) \rangle \neq 0$, which is a contradiction.

The lemma implies the following about PCA with outliers.

▶ **Theorem 12.** *Suppose δ, ϵ are constants such that SSE(δ, ϵ) is hard. Then, even in the zero error case of PCA with outliers, there is no efficient algorithm that can find a subspace of dimension $k/6\sqrt{\epsilon}$ containing all but $(1 + \delta/4)m$ of the points. (Recall m is the prescribed number of outliers.)*

The theorem says that even if we allow a $(1 + \delta)$ fraction more outliers, we cannot have any constant factor approximation, assuming SSE. This, in a sense, is why the dimension of the space returned in Theorem 2 has to have a dependence on δ.

Proof. We consider the reduction as above, and set the upper bound on the number of outliers to

$$m := \frac{n\Delta}{2} - \frac{\delta n \Delta}{2} = \frac{n\Delta}{2}\left(1 - \frac{\delta}{2}\right).$$

In the YES case of SSE, as discussed above, the space spanned by the standard basis vectors corresponding to the non-expanding set contains all the edges incident to that set, and hence we obtain the desired bound on the number of outliers.

In the NO case, by the choice of parameters, violating the number of outliers by a factor $(1 + \delta/4)$ still means that we must have at least $\Delta \cdot n\delta/8$ *inlier* columns. Suppose there is a space of dimension $\leq c\delta n$ that contains this many columns. Then, by Lemma 11, there must be a set of $2c\delta n$ that has at least $\Delta \cdot n\delta/8$ edges. Now a uniformly random subset of δn of these vertices has (in expectation) at least $\Delta \cdot n\delta/32c^2$ edges. Now, if every set of size δn has expansion at least $1 - \epsilon$ (since we are in the NO case), then every such set contains at most $\epsilon\Delta \cdot n\delta$ edges. Thus, we have that

$$\frac{1}{32c^2} \leq \epsilon, \quad \text{which implies } c \geq \frac{1}{6\sqrt{\epsilon}}.$$

This completes the proof of the theorem. ◀

5.2 Reduction from smallest r-edge subgraph

If we do not allow *any* slack in the number of outliers, we show that the situation is rather hopeless. The smallest r-edge subgraph (SrES) [10] problem is the following: given a graph $G = (V, E)$ on n vertices, and a parameter $r \leq |E|$, the goal is to find an induced subgraph H with the smallest number of *vertices*, that has at least r edges. The problem is closely related to (and in some sense is a *dual* of) the densest k-subgraph problem (DkS) [2]. [10] obtained

the best known approximation algorithms for the SrES problem, with an approximation factor $O(n^{2-\sqrt{3}+\epsilon})$, for any $\epsilon > 0$. The complexity of the problem is also closely related to that of DkS, and indeed, the following is believed to be true (see [10, 3] and references therein):

▶ **Conjecture 1.** *The smallest r-edge subgraph problem is NP-hard to approximate to a factor n^c, for some absolute constant $c > 0$. Specifically, there exist parameters r, d such that it is NP-hard to distinguish between*

YES: there exists an induced subgraph on d vertices with r edges.
NO: the smallest induced subgraph containing r edges has at least dn^c vertices.

Now, the following is a simple corollary.

▶ **Corollary 13.** *Consider the PCA with outliers problem in which the algorithm is constrained to ignore at most m outliers (without any slack). Then, assuming Conjecture 1, there is a constant $c > 0$ such that it is NP-hard to find a kn^c dimensional subspace that results in a multiplicative approximation to the objective.*

Proof. We can use the same argument as before, and give a reduction from a gap version of SrES. If we set the number of outliers to be $|E| - r$, then in the YES case, there is a subspace of dimension d containing r edges, while in the NO case, any such subspace must have a dimension at least $dn^c/2$ (using Lemma 11). This completes the proof. ◀

References

1 Maria-Florina Balcan, Avrim Blum, and Santosh Vempala. A discriminative framework for clustering via similarity functions. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, STOC '08, pages 671–680, New York, NY, USA, 2008. ACM. doi:10.1145/1374376.1374474.

2 Aditya Bhaskara, Moses Charikar, Eden Chlamtac, Uriel Feige, and Aravindan Vijayaraghavan. Detecting high log-densities: an $n^{1/4}$ approximation for densest k-subgraph. In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA*, pages 201–210, 2010. doi:10.1145/1806689.1806718.

3 Aditya Bhaskara, Moses Charikar, Venkatesan Guruswami, Aravindan Vijayaraghavan, and Yuan Zhou. Polynomial integrality gaps for strong sdp relaxations of densest k-subgraph. In *ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan*, 2012. doi:10.1145/1806689.1806718.

4 S. Charles Brubaker. Robust PCA and clustering in noisy mixtures. In *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2009, New York, NY, USA, January 4-6, 2009*, pages 1078–1087, 2009.

5 Emmanuel J. Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *J. ACM*, 58(3):11:1–11:37, jun 2011. doi:10.1145/1970392.1970395.

6 Moses Charikar, Samir Khuller, David M. Mount, and Giri Narasimhan. Algorithms for facility location problems with outliers. In *Proceedings of the Twelfth Annual Symposium on Discrete Algorithms, January 7-9, 2001, Washington, DC, USA.*, pages 642–651, 2001. URL: http://dl.acm.org/citation.cfm?id=365411.365555.

7 Moses Charikar, Jacob Steinhardt, and Gregory Valiant. Learning from untrusted data. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, pages 47–60, New York, NY, USA, 2017. ACM. doi:10.1145/3055399.3055491.

8 Ke Chen. A constant factor approximation algorithm for k-median clustering with outliers. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '08, pages 826–835, Philadelphia, PA, USA, 2008. Society for Industrial and Applied Mathematics. URL: `http://dl.acm.org/citation.cfm?id=1347082.1347173`.

9 Flavio Chierichetti, Sreenivas Gollapudi, Ravi Kumar, Silvio Lattanzi, Rina Panigrahy, and David P. Woodruff. Algorithms for lp low-rank approximation. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 806–814, 2017. URL: `http://proceedings.mlr.press/v70/chierichetti17a.html`.

10 Eden Chlamtac, Michael Dinitz, and Robert Krauthgamer. Everywhere-sparse spanners via dense subgraphs. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 758–767, 2012. `doi:10.1109/FOCS.2012.61`.

11 Amit Deshpande and Santosh Vempala. Adaptive sampling and fast low-rank matrix approximation. In Josep Díaz, Klaus Jansen, José D. P. Rolim, and Uri Zwick, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 292–303, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

12 I. Diakonikolas, G. Kamath, D. M. Kane, J. Li, A. Moitra, and A. Stewart. Robust estimators in high dimensions without the computational intractability. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 655–664, Oct 2016. `doi:10.1109/FOCS.2016.85`.

13 Ilias Diakonikolas, Daniel M. Kane, and Alistair Stewart. List-decodable robust mean estimation and learning mixtures of spherical gaussians. *CoRR*, abs/1711.07211, 2017. `arXiv:1711.07211`.

14 Uriel Feige and Joe Kilian. Heuristics for semirandom graph problems. *Journal of Computing and System Sciences*, 63:639–671, 2001.

15 Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981. `doi:10.1145/358669.358692`.

16 Alan M. Frieze and Ravi Kannan. Quick approximation to matrices and applications. *Combinatorica*, 19(2):175–220, 1999. `doi:10.1007/s004930050052`.

17 Gene H. Golub and Charles F. Van Loan. *Matrix Computations (3rd Ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 1996.

18 Shalmoli Gupta, Ravi Kumar, Kefu Lu, Benjamin Moseley, and Sergei Vassilvitskii. Local search methods for k-means with outliers. *Proc. VLDB Endow.*, 10(7):757–768, 2017. `doi:10.14778/3067421.3067425`.

19 V. Guruswami and A. K. Sinop. Optimal column-based low-rank matrix reconstruction. In *SODA*, 2012.

20 Moritz Hardt and Ankur Moitra. Algorithms and hardness for robust subspace recovery. In *COLT 2013 - The 26th Annual Conference on Learning Theory, June 12-14, 2013, Princeton University, NJ, USA*, pages 354–375, 2013. URL: `http://jmlr.org/proceedings/papers/v30/Hardt13.html`.

21 Samuel B. Hopkins and Jerry Li. Mixture models, robustness, and sum of squares proofs. *CoRR*, abs/1711.07454, 2017. `arXiv:1711.07454`.

22 Peter J. Huber. *Robust Statistics*. Wiley, 1981.

23 Peter J. Huber and Elvezio M. Ronchetti. *Robust Statistics, 2nd Edition*. Wiley, 2009.

24 Pravesh K. Kothari and Jacob Steinhardt. Better agnostic clustering via relaxed tensor norms. *CoRR*, abs/1711.07465, 2017. `arXiv:1711.07465`.

25 Ravishankar Krishnaswamy, Shi Li, and Sai Sandeep. Constant approximation for k-median and k-means with outliers via iterative rounding. *CoRR*, abs/1711.01323, 2017.

26 K. A. Lai, A. B. Rao, and S. Vempala. Agnostic estimation of mean and covariance. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 665–674, Oct 2016. doi:10.1109/FOCS.2016.76.

27 Konstantin Makarychev, Yury Makarychev, and Aravindan Vijayaraghavan. Learning communities in the presence of errors. In *Proceedings of the 29th Conference on Learning Theory, COLT 2016, New York, USA, June 23-26, 2016*, pages 1258–1291, 2016.

28 Frank McSherry. Spectral partitioning of random graphs. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001, Las Vegas, Nevada, USA*, pages 529–537, 2001. doi:10.1109/SFCS.2001.959929.

29 Prasad Raghavendra and David Steurer. Graph expansion and the unique games conjecture. In Leonard J. Schulman, editor, *STOC*, pages 755–764. ACM, 2010. doi:10.1145/1806689.1806788.

30 Jacob Steinhardt, Moses Charikar, and Gregory Valiant. Resilience: A criterion for learning in the presence of arbitrary outliers. In *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, pages 45:1–45:21, 2018. doi:10.4230/LIPIcs.ITCS.2018.45.

31 G. W. Stewart and Ji guang Sun. *Matrix Perturbation Theory*. Academic Press, 1990.

32 H. Xu, C. Caramanis, and S. Mannor. Outlier-robust pca: The high-dimensional case. *IEEE Transactions on Information Theory*, 59(1):546–572, Jan 2013. doi:10.1109/TIT.2012.2212415.

33 H. Xu, C. Caramanis, and S. Sanghavi. Robust pca via outlier pursuit. *IEEE Transactions on Information Theory*, 58(5):3047–3064, May 2012. doi:10.1109/TIT.2011.2173156.

34 Xinyang Yi, Dohyung Park, Yudong Chen, and Constantine Caramanis. Fast algorithms for robust PCA via gradient descent. *CoRR*, abs/1605.07784, 2016. arXiv:1605.07784.

A Guessing the optimum

In all our algorithms, we assume that we have a guess for the optimum error, up to a multiplicative factor of $(1+\epsilon)$. Note that the error is $\|B - B_k\|_F^2$, as defined in the introduction. We now argue that we can always obtain a polynomial number of guesses, one of which is accurate. This follows immediately if we can show that the error lies in a range $[L, U]$, where U/L is at most $\exp(\text{poly}(n, b))$, where b is the total bit complexity of the input. (This is because we can discretize the range into $\text{poly}(n, b)/\epsilon$ intervals using multiples of $(1+\epsilon)$.)

This is not immediate in our setting because $\|B - B_k\|_F^2$ can actually be zero. But we can show that if the error is *non-zero*, it can be lower bounded by $\exp(-\text{poly}(n, b))$. Such results are quite well-known (see [17]), but we sketch a short proof below.

Exponential range for the optimum. The key claim is that if a $d \times k$ matrix C has linearly independent columns, it has $\sigma_{\min} \geq \exp(-\text{poly}(n, b))$, where b is the total bit complexity of C. This can be proved as follows. Note that $\sigma_{\min}$ is the smallest eigenvalue of $C^T C$. Now the characteristic polynomial of $C^T C$ has coefficients that are all at most $\exp(\text{poly}(n, b))$, as they are appropriate determinants. Suppose the polynomial is $\sum_{i=0}^{k} a_i \lambda^i$. By linear independence, we know that $a_0 \neq 0$. Further, a_0 is the determinant of $C^T C$, and hence is bounded from below by $\exp(-\text{poly}(n, b))$; this is because the sum of a set of numbers of a certain precision is either zero or is at least the "least count" of that precision.

Now, the magnitude of the smallest non-zero real root of any polynomial can be bounded from below by $|a_0|/(|a_0| + |a_1| + \cdots + |a_d|)$. As the a_i are all at most $\exp(\text{poly}(n, b))$, the desired conclusion follows.

Greedy Bipartite Matching in Random Type Poisson Arrival Model

Allan Borodin
University of Toronto, 10 Kings College Road, Toronto, Canada
bor@cs.toronto.edu

Christodoulos Karavasilis
University of Toronto, 10 Kings College Road, Toronto, Canada
ckar@cs.toronto.edu

Denis Pankratov
University of Toronto, 10 Kings College Road, Toronto, Canada
denisp@cs.toronto.edu

Abstract

We introduce a new random input model for bipartite matching which we call the Random Type Poisson Arrival Model. Just like in the known i.i.d. model (introduced by Feldman et al. [6]), online nodes have types in our model. In contrast to the adversarial types studied in the known i.i.d. model, following the random graphs studied in Mastin and Jaillet [1], in our model each type graph is generated randomly by including each offline node in the neighborhood of an online node with probability c/n independently. In our model, nodes of the same type appear consecutively in the input and the number of times each type node appears is distributed according to the Poisson distribution with parameter 1. We analyze the performance of the simple greedy algorithm under this input model. The performance is controlled by the parameter c and we are able to exactly characterize the competitive ratio for the regimes $c = o(1)$ and $c = \omega(1)$. We also provide a precise bound on the expected size of the matching in the remaining regime of constant c. We compare our results to the previous work of Mastin and Jaillet who analyzed the simple greedy algorithm in the $G_{n,n,p}$ model where each online node type occurs exactly once. We essentially show that the approach of Mastin and Jaillet can be extended to work for the Random Type Poisson Arrival Model, although several nontrivial technical challenges need to be overcome. Intuitively, one can view the Random Type Poisson Arrival Model as the $G_{n,n,p}$ model with less randomness; that is, instead of each online node having a new type, each online node has a chance of repeating the previous type.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis, Theory of computation → Online algorithms

Keywords and phrases bipartite matching, stochastic input models, online algorithms, greedy algorithms

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.5

Related Version `https://arxiv.org/abs/1805.00578`

Funding Research is supported by NSERC.

Acknowledgements We thank the anonymous reviewers of APPROX/RANDOM 2018, whose valuable comments helped us improve the presentation of the paper.

1 Introduction

Online bipartite matching is a problem with a wide variety of applications and has received significant attention after the seminal work of Karp, Vazirani and Vazirani [10] who showed an optimal $(1 - 1/e)$ randomized algorithm for the unweighted case in the adversarial input model. Applications such as internet advertising (see the excellent survey by Mehta [13] and references therein) and job allocation have given rise to problems that are naturally described as bipartite matching problems and this has been a strong motivation for developing better algorithms. Most of the work in online bipartite matching is with respect to a vertex arrival model where vertices on one side of the bipartite graph are known to the algorithm in advance and vertices on the other side are revealed online along with their adjacent vertices.

Adversarial models are, of course, pessimistic in terms of performance results. For many applications, more realistic assumptions will lead to significantly improved results. In this regard, various stochastic input models have been proposed and analyzed. These include the random order model, known and unknown i.i.d. distribution models, and the Erdős-Rényi random graph model. We propose a new model (see Section 2) that is closely related to both the i.i.d. model and the Erdős-Rényi model. The motivation for our model is that in various applications (e.g. document streams [11] and internet traffic [15]), one often observes the bursty nature of inputs. One can also think of this as a very restricted form of a Markov chain model.

Whereas previous work in the i.i.d. models take advantage of the input distributions in terms of the decisions being made (i.e. as to how to match the online vertices), we follow the work of Mastin and Jaillet [1] who study the performance of the simplest greedy algorithm (which we will call Greedy) that always matches (when possible) an online vertex to the first (in some fixed ordering of the offline vertices) unmatched adjacent neighbor[1]. We note that this simple greedy algorithm does not utilize any information about the nature of the input sequence (including the number of online vertices).

We know that theoretically (i.e. in terms of the expected approximation ratio), that knowledge of the distribution and/or randomization in the algorithm will allow for online algorithms with improved performance. However, in simulations with respect to various stochastic models, we find that the simplest deterministic greedy algorithm is competitive with more specialized algorithms [5]. We also note that when a type graph is chosen adversarially in the i.i.d. model, Goel and Mehta [7] show that the approximation ratio of Greedy is precisely $1 - 1/e \approx .632$.

Mastin and Jaillet show that the approximation ratio of Greedy in the Erdős-Rényi model is at least 0.837. Intuitively, our model introduces correlations between consecutive online nodes. This causes significant technical difficulties in the analysis that we are able to overcome. In addition, the correlations do not seem favourable to the Greedy algorithm, and it is natural to expect Greedy to have worse performance in our model than in the Erdős-Rényi model. Our results confirm this intuition. Our input model has a parameter c that controls the expected degree of online nodes. Similar to the work of Mastin and Jaillet [1], we analyze the performance of Greedy in three different regimes of c: $c = o(1)$, $c = \omega(1)$, and constant c. We compute the exact asymptotic fraction of offline nodes matched by Greedy in each of these regimes. We also show how to derive an upper bound on the

[1] In the version of Greedy that is studied by Mastin and Jaillet, an online vertex is matched to a randomly chosen available neighbor. Later, we shall see that for our input model, as well as $G_{n,n,p}$, the two versions of Greedy have the same approximation ratio, since they result in the same stochastic process governing the change in the number of matched offline nodes when an online node arrives.

size of a maximum matching in our model from the existing upper bounds on the size of a maximum matching in the Erdős-Rényi model. Combining the two results, we derive a lower bound on the approximation ratio of Greedy in all regimes of c. Minimizing this lower bound over c, we find that Greedy has approximation ratio ≥ 0.715 for all values of c in our model.

Organization. The rest of the paper is organized as follows. Section 2 describes our model and different views of it, as well as some background material needed for the rest of the paper. Section 3 is the technical core of the paper and contains the analysis of Greedy in the three regimes of c: $c = o(1)$, $c = \omega(1)$, and $c = \Theta(1)$. Section 4 concludes the paper with a brief discussion of various input models and some open questions.

2 Preliminaries

We shall consider bipartite graphs $G = (U, V, E)$ in the vertex arrival model. The vertices in U are referred to as the offline vertices and are known to an algorithm in advance. The vertices in V are referred to as online vertices and arrive one at a time. When a vertex from V is revealed, all its neighbors in U are revealed as well. A simple greedy algorithm matches the arriving vertex $v \in V$ to the first available neighbor in U (if there is one, according to some fixed ordering). We shall denote this algorithm by Greedy. We consider the behavior of Greedy on specific families of random graphs generated by, what we call, the Random Type Poisson Arrival Model. This model has two parameters: $n \in \mathbb{N}$ which is equal to $|U|$ and intuitively measures the size of the instance, and $c \in \mathbb{R}_{\geq 0}$ which controls the density of edges. The random graph in our model is generated as follows: the offline nodes U are set to be $[n]$. Online nodes are generated iteratively and randomly using the following process. For each $i \in [n]$ generate a random type i by including each offline node $j \in [n]$ in the neighborhood of the type with probability c/n independently. Then we sample Z_i from the Poisson distribution with parameter 1 and generate Z_i online nodes of type i, and continue. We shall denote a graph G distributed according to the Random Type Poisson Arrival Model with parameters n and c as $G \sim \text{RTPAM}(n, c)$. Our model can be viewed in different ways. We refer to the current view of the RTPAM(n, c) model as the "one-step view." Next we describe another view.

The bipartite Erdős-Rényi model is denoted by $G_{n,n,p}$. The random graph $G = (U, V, E)$ in the $G_{n,n,p}$ model is generated as follows: $U = V = [n]$, and each edge $\{i, j\}$ is included in G with probability p independently. Our RTPAM(n, c) model can be alternatively viewed as a two-step process. At first, we generate a *type graph* $\widehat{G} = (U, V, E)$ from the $G_{n,n,p}$ distribution where $p = c/n$. For each type $i \in V$ we sample $Z_i \sim \text{Poi}(1)$. Secondly, an *instance graph* is created by keeping the same U as in the type graph, and replicating each type i node Z_i times (note that this means removing type i when $Z_i = 0$). In the rest of the paper, we shall freely switch between the two different views of the RTPAM(n, c) model. Thus, we shall occasionally refer to the type graph of the RTPAM(n, c) graph. We refer to this alternative view of the RTPAM(n, c) model as the "two-step view."

Intuitively, in the one-step view, Z_i is drawn from the Poisson distribution in an online fashion whereas in the two-step view, Z_i is drawn initially. It should be clear that these views do not change the model. However, our proofs are facilitated by having these different views.

We shall measure the performance of an algorithm in two ways: in terms of the asymptotic approximation ratio, and in terms of the fraction of the matched offline nodes.

▶ **Definition 1.** Let ALG be a deterministic online algorithm solving the bipartite matching problem over random graphs G_n parameterized by the input size n. We write $\mathrm{ALG}(G_n)$ to denote the size of the matching (random variable) that is constructed by running ALG on G_n. We write $\mathrm{OPT}(G_n)$ to denote the size of a maximum matching in G_n. *The asymptotic approximation ratio* of ALG with respect to G_n is defined as:

$$\rho(\mathrm{ALG}, G_n) = \liminf_{n\to\infty} \frac{\mathbb{E}(\mathrm{ALG}(G_n))}{\mathbb{E}(\mathrm{OPT}(G_n))}.$$

The fraction of matched offline nodes of ALG with respect to G_n is defined as:

$$\mu(\mathrm{ALG}, G_n) = \liminf_{n\to\infty} \frac{\mathbb{E}(\mathrm{ALG}(G_n))}{n}.$$

We shall use the following notation for the well-known distributions:

▶ **Definition 2.**

- $\mathrm{Poi}(\lambda)$ – the Poisson distribution with parameter λ,
- $\mathrm{Bin}(n, p)$ – the Binomial distribution with parameters $n \in \mathbb{Z}_{\geq 0}$ (number of trials) and $p \in [0, 1]$ (probability of success)
- $\mathrm{Ber}(p)$ – the Bernoulli distribution with parameter p.

We shall also write $\mathrm{Poi}(\lambda)$, $\mathrm{Bin}(n, p)$, etc., as a placeholder for a random variable distributed according to the corresponding distribution. This is done to simplify the notation when the name of the random variable is not important and only the parameters of the distribution are of interest.

▶ **Definition 3.** We write $\mathbb{I}(E)$ to denote the indicator random variable for an event E.

In the paper, we show how to compute $\mu(\textsc{Greedy}, \mathrm{RTPAM}(n, c))$ exactly. In order to provide a bound on the asymptotic approximation ratio of Greedy we also need to know the size of a maximum matching in $\mathrm{RTPAM}(n, c)$. The size of a maximum matching is not known even in $G_{n,n,c/n}$ model, but there is a known upper bound due to Bollobás and Brightwell that we will be able to use to derive a lower bound on the asymptotic approximation ratio of Greedy in the $\mathrm{RTPAM}(n, c)$ model.

▶ **Theorem 4** ([4])**.**

$$\frac{\mathbb{E}(\mathrm{OPT}(G_{n,n,c/n}))}{n} \leq 2 - \frac{\gamma^* + \gamma_* + \gamma^*\gamma_*}{c} + o(1),$$

where γ_* is the smallest solution to the equation $x = c\exp(-c\exp(-x))$ and $\gamma^* = c\exp(-\gamma_*)$. See Figure 4 for the shape of this upper bound as a function of c.

We shall later compare our results with the following result giving the performance of Greedy in the $G_{n,n,c/n}$ model due to Mastin and Jaillet.

▶ **Theorem 5** ([1])**.** *For $c \in \mathbb{R}_{>0}$ we have*

$$\mu(\textsc{Greedy}, G_{n,n,c/n}) = 1 - \frac{\log(2 - e^{-c})}{c}.$$

3 Greedy Matching in Random Type Poisson Arrival Model

3.1 The Regime of $c = o(1)$

In this section we show that in expectation Greedy finds an almost-maximum matching in RTPAM(n, c) when $c = o(1)$. The high level idea is to consider the two-step view of RTPAM(n, c) and observe that when $c = o(1)$ most of the type graph consists of *isolated* edges – edges with both endpoints of degree 1. Thus, no matter how many times an online node corresponding to an isolated edge is generated in the instance graph, both Greedy and OPT can match it exactly once (given that it is generated at all). The expected number of the non-isolated edges is of smaller order of magnitude than the expected number of isolated edges and can be ignored for the purpose of computing the asymptotic approximation ratio.

▶ **Theorem 6.** *Let $c = o(1)$ then we have:*

$$\rho(\text{Greedy}, \text{RTPAM}(n, c)) = 1.$$

Proof. (similar to the proof of Lemma 2 in [1]) Set $p = c/n = o(1/n)$, and consider the two-step view of RTPAM(n, c). The probability that an edge between i and j appears and is isolated in *the type graph* is:

$$\Pr(\{i, j\} \text{ is isolated in the type graph }) = p(1-p)^{2n-2}.$$

Observe that if type j is generated at least once in *the instance graph*, i.e., $Z_j \geq 1$, and $\{i, j\}$ is an isolated edge in *the type graph* then Greedy will include $\{i, j\}$ in the matching exactly once. In addition, observe that the events $Z_j \geq 1$ and "$\{i, j\}$ is isolated in the type graph" are independent. Let $W_{i,j}$ be a random variable indicating whether $\{i, j\}$ is included by Greedy. Then, we have:

$$\Pr(W_{i,j} = 1) \geq \Pr(Z_j \geq 1) \Pr(\{i, j\} \text{ is isolated in the type graph}) = \left(1 - \frac{1}{e}\right) p(1-p)^{2n-2}.$$

Let M be the matching produced by Greedy. We have $|M| = \sum_{i,j} W_{i,j}$. It follows that:

$$\mathbb{E}(|M|) = \mathbb{E}\left(\sum_{i,j} W_{i,j}\right) = \sum_{i,j} \Pr(W_{i,j} = 1) \geq n^2 \left(1 - \frac{1}{e}\right) p(1-p)^{2n-2}.$$

Let Q_j denote the number of neighbors of a node of type j. Observe that in the instance graph the nodes corresponding to type j can be matched in an optimal matching at most $Q_j \mathbb{I}(Z_j \geq 1)$ times. Note that Q_j and Z_j are independent. Let M^* denote a maximum matching in the instance graph. Then we have

$$\mathbb{E}(|M^*|) \leq \sum_j \mathbb{E}(Q_j \mathbb{I}(Z_j \geq 1)) = \sum_j np(1 - 1/e) = n^2 p(1 - 1/e).$$

Combining this with the above lower bound on $\mathbb{E}(|M|)$ we get

$$\frac{\mathbb{E}(|M|)}{\mathbb{E}(|M^*|)} \geq (1-p)^{2n-2} = \left(e^{-o(\frac{1}{n})}\right)^{2n-2} = e^{-o(1)} \xrightarrow[n\to\infty]{} 1. \qquad \blacktriangleleft$$

3.2 The Regime of Constant c

Fix a constant $c \in \mathbb{R}_{>0}$. We can view Greedy as constructing the matching in rounds. Consider the one-step view of RTPAM(n, c). During round i, a new type is created and Z_i nodes corresponding to that type are generated. We let Y_i denote the number of online nodes matched by Greedy by the beginning of round i. We also let X_i denote the number of neighbors of type i that were *not* matched in any of the earlier rounds. In this section, we show how to compute the asymptotic fraction of matched offline nodes by Greedy *exactly*. More specifically, we derive an asymptotically accurate (implicit) expression for Y_n and show how to compute it for each value of c. In addition, we show that existing upper bounds on the maximum matching in the $G_{n,n,c/n}$ model carry over to the RTPAM(n, c) model. This allows us to derive lower bounds on the *competitive ratio* of Greedy in RTPAM(n, c).

High level idea. We use the method of partial differential equations (see, e.g., [19, 18, 12]) to derive the asymptotic behavior of Y_n. The goal is to write the expression $\mathbb{E}(Y_{i+1} - Y_i \mid Y_i)$ in terms of Y_i/n, i.e., $\mathbb{E}(Y_{i+1} - Y_i \mid Y_i) = f_c(Y_i/n)$ for some "simple" function f_c. This gives us a difference equation for Y_i. Now, pretend that there is a function $g_c : [0, 1] \to [0, 1]$ that gives a good approximation to Y_i, i.e., $g_c(t) \approx Y_{tn}/n$ for $t \in [0, 1]$. Consider a syntactic replacement of the difference equation for Y_i with a differential equation for g_c, i.e., $g_c'(t) = f_c(g_c(t))$. In addition, set the correct initial value condition $g_c(0) = Y_0 = 0$. The differential equation method allows us to conclude that under a mild condition on f_c (namely, being Lipschitz), the solution g_c is unique and asymptotically converges to Y_n/n, i.e., $Y_n = g_c(1)n + o(n)$. In particular, we have $\mu(\text{Greedy}, \text{RTPAM}(n, c)) = g_c(1)$. In our setting, we will see that it is not clear how to write $\mathbb{E}(Y_{i+1} - Y_i \mid Y_i)$ as a function of Y_i/n. It turns out that the method still works as long as $\mathbb{E}(Y_{i+1} - Y_i \mid Y_i)$ is close to $f_c(Y_i/n)$ in the following sense: $\lim_{n\to\infty} |\mathbb{E}(Y_{i+1} - Y_i \mid Y_i) - f_c(Y_i/n)| = 0$. This is precisely what we do in this section.

The number of nodes matched by Greedy in round i is exactly equal[2] to $\min(X_i, Z_i)$. By the definition of the RTPAM(n, c) model, we have $X_i \sim \text{Bin}(n - Y_i, c/n)$. Therefore, we have

$$\mathbb{E}(Y_{i+1} - Y_i \mid Y_i) = \mathbb{E}(\min(\text{Bin}(n - Y_i, c/n), \text{Poi}(1)) \mid Y_i). \tag{1}$$

Unfortunately, as mentioned above this expectation does not seem to have a nice form and we do not know how to set up an associated differential equation. Instead, we shall approximate the difference equation $\mathbb{E}(Y_{i+1} - Y_i \mid Y_i)$ by another expression that is easier to handle. Those familiar with the Poisson limit theorem will immediately recognize the following as the most natural choice:

$$\mathbb{E}(\min(\text{Bin}(n - Y_i, c/n), \text{Poi}(1)) \mid Y_i) \approx \mathbb{E}(\min(\text{Poi}(c(1 - Y_i/n)), \text{Poi}(1)) \mid Y_i).$$

We need to analyze how accurate this approximation is, but first we show how to derive a relatively simple expression for the right hand side. Define $h(x) := \mathbb{E}(\min(\text{Poi}(x), \text{Poi}(1)))$. Although the function $h(x)$ does not have a closed-form expression in terms of widely-known functions such as $\sin, \cos, \exp$, etc., it does have a closed-form expression in terms of the modified Bessel functions of the first kind and the Marcum's Q functions:

[2] Observe that this holds in all regimes of c and that this expression remains the same if we consider the version of Greedy of Mastin and Jaillet, where an online node is matched to a *random* available neighbor. Therefore, the exact version of Greedy does not matter for our input model. Similar argument applies to $G_{n,n,p}$.

▶ **Definition 7.** The modified Bessel functions of the first kind are defined as follows:

$$I_k(x) = \sum_{i=0}^{\infty} \frac{1}{i!\Gamma(i+k+1)} \left(\frac{x}{2}\right)^{2i+k}.$$

For an integer $k \geq 0$, it becomes $I_k(x) = \sum_{i=0}^{\infty} \frac{1}{i!(i+k)!} \left(\frac{x}{2}\right)^{2i+k}$. We note that the modified Bessel functions have the following symmetry property: $I_k(x) = I_{-k}(x)$.

▶ **Definition 8.** Marcum's Q function is defined as follows:

$$Q_n(a,b) = \exp\left(-\frac{a^2+b^2}{2}\right) \sum_{k=1-n}^{\infty} \left(\frac{a}{b}\right)^k I_k(ab).$$

Now, the closed-form expression for $h(x)$ can be derived from the answer on Stats Stackexchange [16]. For completeness, we reproduce the derivation in Appendix A.

▶ **Lemma 9** ([16]). *For $x > 0$ we have*

$$h(x) = \frac{1 + x - 2e^{-x-1}\left(I_0(2\sqrt{x}) + \sqrt{x}I_1(2\sqrt{x})\right) - (1-x)\left(1 - 2Q_1(\sqrt{2x}, \sqrt{2})\right)}{2}.$$

Thus, in terms of our overview we have $f_c(x) = h(c(1-x))$, because we hope to show that

$$\mathbb{E}(\min(\mathrm{Bin}(n-Y_i, c/n), \mathrm{Poi}(1)) \mid Y_i) \approx \mathbb{E}(\min(\mathrm{Poi}(c(1-Y_i/n)), \mathrm{Poi}(1)) \mid Y_i) = h(c(1-Y_i/n)).$$

To apply the method of differential equations we need to analyze the above approximation and show that f_c is Lipschitz. We start by analyzing how good the approximation is.

▶ **Lemma 10.**

$$\lim_{n\to\infty} \left|\mathbb{E}(\min(\mathrm{Bin}(n-Y_i, c/n), \mathrm{Poi}(1)) \mid Y_i) - h(c(1-Y_i/n))\right| = 0.$$

Proof. We introduce the following useful notation: $N = n - Y_i$, $p = c/n$, and $\lambda = c(1-Y_i/n)$. Moreover, we let $b_r = \Pr(\mathrm{Bin}(N,p) = r)$ and $p_r(a) = \Pr(\mathrm{Poi}(a) = r)$. Let $W_i \sim \mathrm{Poi}(c(1 - Y_i/n))$. Then the statement of the lemma can be translated into

$$\left|\mathbb{E}(\min(X_i, Z_i) - \min(W_i, Z_i) \mid Y_i)\right|$$

The expectation is just a big sum. Let's consider individual terms and their contributions to the overall sum.

1. **Case: $W_i < X_i < Z_i$.** The contribution of $X_i = k, W_i = j, Z_i = \ell$ when $j < k < \ell$ is $k - j$.
2. **Case: $X_i < W_i < Z_i$.** The contribution of $X_i = j, W_i = k, Z_i = \ell$ when $j < k < \ell$ is $j - k = -(k - j)$.
3. **Case: $W_i < Z_i \leq X_i$.** The contribution of $X_i = k, W_i = j, Z_i = \ell$ when $j < \ell$ and $k \geq \ell$ is $\ell - j$.
4. **Case: $X_i < Z_i \leq W_i$.**The contribution of $X_i = j, W_i = k, Z_i = \ell$ when $j < \ell$ and $k \geq \ell$ is $j - \ell$.
5. **Cases: (a) $X_i = W_i$; (b) $Z_i \leq X_i, W_i$.** The contribution is 0.

We pair up terms corresponding to (1) with (2) and terms corresponding to (3) with (4). Define

$$S_1 = \sum_{k,j,\ell=0}^{\infty} \mathbb{I}(j < k < \ell) \left((k-j)b_k p_j(\lambda)p_\ell(1) + (j-k)b_j p_k(\lambda)p_\ell(1)\right),$$

and

$$S_2 = \sum_{k,j,\ell=0}^{\infty} \mathbb{I}(j < \ell, k \geq \ell) \left((\ell-j)b_k p_j(\lambda)p_\ell(1) + (j-\ell)b_j p_k(\lambda)p_\ell(1)\right).$$

Thus, we get that

$$\left|\mathbb{E}(\min(X_i, Z_i) - \min(W_i, Z_i) \mid Y_i)\right| = |S_1 + S_2| \leq |S_1| + |S_2|.$$

We will show that $\lim_{n\to\infty} |S_1| = 0$. Similar argument implies that $\lim_{n\to\infty} |S_2| = 0$. We have

$$\begin{aligned} S_1 &= \sum_{k,j,\ell=0}^{\infty} \mathbb{I}(j < k < \ell) p_\ell(1)(k-j)(b_k p_j(\lambda) - b_j p_k(\lambda)) \\ &= \sum_{k,j,\ell=0}^{\infty} \mathbb{I}(j < k < \ell) p_\ell(1)(k-j)((b_k - p_k(\lambda))p_j(\lambda) - (b_j - p_j(\lambda))p_k(\lambda)) \\ &= S_1^1 + S_1^2, \end{aligned}$$

where $S_1^1 = \sum_{k,j,\ell=0}^{\infty} \mathbb{I}(j < k < \ell) p_\ell(1)(k-j)(b_k - p_k(\lambda))p_j(\lambda)$ and $S_1^2 = S_1 - S_1^1$. Again, $|S_1| \leq |S_1^1| + |S_1^2|$. We will show that $\lim_{n\to\infty} |S_1^1| = 0$, and a similar argument implies that the same holds for $|S_1^2|$. We have

$$|S_1^1| \leq \sum_{k=0}^{\infty} \left(\sum_{j,\ell=0}^{\infty} \mathbb{I}(j < k < \ell)(k-j) p_\ell(1) p_j(\lambda) \right) |b_k - p_k(\lambda)|.$$

By [14], we have $\lim_{n\to\infty} |S_1^1| = 0$ if and only if

$$\sum_{k,j,\ell=0}^{\infty} \mathbb{I}(j < k < \ell)(k-j) p_\ell(1) p_j(\lambda) p_k(\lambda) = O(1).$$

Lastly, we have

$$\begin{aligned} &\sum_{k,j,\ell=0}^{\infty} \mathbb{I}(j < k < \ell)(k-j) p_\ell(1) p_j(\lambda) p_k(\lambda) \leq \sum_{k,j,\ell=0}^{\infty} \mathbb{I}(j < k < \ell) k p_\ell(1) p_j(\lambda) p_k(\lambda) \\ &\leq \sum_{k,j,\ell=0}^{\infty} p_\ell(1) p_j(\lambda) k p_k(\lambda) \leq \lambda \xrightarrow[n\to\infty]{} c = O(1). \end{aligned}$$

◀

Due to space considerations we prove that f_c is Lipschitz in Appendix A.

▶ **Lemma 11.** *The function $f_c(x)$ is Lipschitz on $[0, 1]$.*

Finally, we have all the necessary ingredients to prove the main theorem of this section. Although this theorem does not give an explicit closed-form expression for $\mu(\textsc{Greedy}, \text{RTPAM}(n, c))$, it gives a simple way to evaluate it numerically for any value of $c > 0$.

▶ **Theorem 12.**

$$\mu(\textsc{Greedy}, \mathrm{RTPAM}(n,c)) = g_c(1),$$

where g_c is a solution to the following differential equation:

$$g_c'(t) = h(c(1-g_c(t))),$$
$$g_c(0) = 0,$$

and h is given in Lemma 9.

Proof. This is a direct application of Wormald's theorem (Theorem 5.1 in [19]) using Lemmas 10 and 11. ◀

Next, we show two upper bounds on $\mathbb{E}(\mathrm{OPT}(\mathrm{RTPAM}(n,c)))$. The minimum of the two will be used to compute the approximation ratio of Greedy.

▶ **Theorem 13.** *For all $c \in \mathbb{R}_{>0}$ we have*

$$\mathbb{E}(\mathrm{OPT}(\mathrm{RTPAM}(n,c))) \le \min\left(nc\left(1-\frac{1}{e}\right), \left(2-\frac{\gamma^*+\gamma_*+\gamma^*\gamma_*}{c}\right)n + o(n)\right),$$

where γ_* is the smallest solution to the equation $x = c\exp(-c\exp(-x))$ and $\gamma^* = c\exp(-\gamma_*)$.

Proof. The first argument in the minimum follows from the proof of Theorem 6. Let Q_j denote the number of neighbors of a node of type j. Observe that the number of nodes of type j participating in any matching is bounded above by $Q_j\mathbb{I}(Z_j \ge 1)$. Using the fact that Q_j and $\mathbb{I}(Z_j \ge 1)$ are independent, taking the expectation and summing over all j results in the upper bound of $nc\left(1-\frac{1}{e}\right)$.

The second argument in the minimum follows from the observation

$$\mathbb{E}(\mathrm{OPT}(\mathrm{RTPAM}(n,c))) \le \mathbb{E}(\mathrm{OPT}(G_{n,n,c/n})) \tag{2}$$

and Theorem 4. Let $\alpha(G)$ denote the independence number of the graph. By Gallai's and Kőnig's theorems it suffices to prove that

$$\mathbb{E}(\alpha(\mathrm{RTPAM}(n,c))) \ge \mathbb{E}(\alpha(G_{n,n,c/n})).$$

Consider the two-step view of $\mathrm{RTPAM}(n,c)$. In the first step, a type graph $\widehat{G} = (U,V,E)$ is generated from the distribution $G_{n,n,c/n}$. Let S be a largest independent set in the type graph. Write $S = S_1 \cup S_2$, where $S_1 = S \cap U$ consists of offline nodes and $S_2 = S \cap V$ consists of online types. In the *instance* graph all nodes from S_1 together with all nodes with types from S_2 will form an independent set. In other words, even if a node of a given type is repeated multiple times from S_2, it can be safely included in an independent set. Thus, we have

$$\begin{aligned}\mathbb{E}(\alpha(\mathrm{RTPAM}(n,c)) \mid S_1, S_2) &\ge \mathbb{E}\left(|S_1| + \sum_{j\in S_2} Z_j \mid S_1, S_2\right)\\ &= |S_1| + \sum_{j\in S_2}\mathbb{E}(Z_j \mid S_1, S_2) = |S_1| + |S_2| = |S|,\end{aligned}$$

where the last equality follows because Z_j is independent of S_1, S_2. Taking the expectation over S proves $\mathbb{E}(\alpha(\mathrm{RTPAM}(n,c))) \ge \mathbb{E}(\alpha(G_{n,n,c/n}))$, since $|S|$ has the same distribution as $\alpha(G_{n,n,c/n})$. ◀

3.3 The Regime of $c = \omega(1)$

In this section we show that GREEDY matches almost all offline nodes in RTPAM(n, c) model when $c = \omega(1)$. Consider the two-step view of RTPAM(n, c). Recall that Z_j refers to the number of nodes of type j that are generated, and that X_j refers to the number of neighbors of a node of type j that have not been matched in any of the previous rounds. Also, recall that GREEDY matches $\mathbb{E}(\min(X_j, Z_j))$ in round j in expectation. We will show that in most rounds $\min(X_j, Z_j)$ is very close to Z_j. This finishes the argument, since $\sum_j \mathbb{E}(Z_j) = n$ is the total number of offline nodes and a trivial upper bound on the size of a maximum matching.

▶ **Theorem 14.** *Let $c = \omega(1)$ then we have:*

$$\rho(\text{GREEDY}, \text{RTPAM}(n, c)) = 1.$$

Proof. Let $p = \frac{c}{n}$ and $k = \frac{n}{\sqrt{c}}$. Fix a round i and assume that at least k offline nodes have *not* been matched in earlier rounds. Then variable X_i has binomial distribution with at least k trials and the probability of success c/n. We will consider $\widetilde{X}_i \sim \text{Bin}(k, c/n)$ such that $\widetilde{X}_i \leq X_i$. We will show that $\Pr(\widetilde{X}_i \geq Z_i) = 1 - o(1)$. Since $Z_i \sim \text{Poi}(1)$ we have:

$$P(Z_i \geq c^{1/100}) = \frac{1}{e} \sum_{j=c^{1/100}}^{\infty} \frac{1}{j!} \leq \frac{1}{c^{1/100}!} \leq \frac{1}{2^{c^{1/100}}}.$$

For $\widetilde{X}_i$ we have $\text{Var}(\widetilde{X}_i) = kp(1-p) = \sqrt{c}(1 - c/n)$ and $\mathbb{E}(\widetilde{X}_i) = \sqrt{c}$. By Chebyshev's inequality

$$\Pr(|\widetilde{X}_i - \sqrt{c}| \geq c^{1/3}) \leq \frac{\sqrt{c}(1 - c/n)}{c^{2/3}} = \frac{1 - c/n}{c^{1/6}}.$$

From these two bounds, it follows that

$$\Pr(\widetilde{X}_i \geq Z_i) \geq \Pr(Y_i \leq c^{1/100} \wedge \widetilde{X}_i \geq \sqrt{c} - c^{1/3}) \geq 1 - \frac{1 - c/n}{c^{1/6}} - \frac{1}{2^{c^{1/100}}} = 1 - o(1).$$

In addition, it is easy to see that in the first $n - 10k$ rounds GREEDY matches at most $n - k$ offline nodes with probability $1 - o(1)$. In particular, the probability of matching more than that is bounded by the probability that $\sum_{i=1}^{n-10k} Z_i > n - k$. Thus, we can condition on having at least k available offline nodes during each of the first $n - 10k$ rounds. Therefore, the expected size of the matching constructed by GREEDY is at least $\sum_{i=1}^{n-10k} \mathbb{E}(\min(Z_i, X_i)) \geq (n - 10k)(1 - o(1)) = n - o(n)$. ◀

3.4 Putting it together

In this section, we take a closer look at our results for GREEDY in RTPAM(n, c) model. We already know that GREEDY achieves competitive ratio 1 in the regimes $c = o(1)$ and $c = \omega(1)$. Hence, we concentrate on the regime of constant c. In Figure 1 we plot the asymptotic fraction of matched offline nodes by GREEDY (Theorem 12) and the upper bound on the fraction of offline nodes in a maximum matching (Theorem 13) as functions of c .

By taking the ratio of the two curves in Figure 1 we obtain a lower bound on the asymptotic approximation ratio of GREEDY in the RTPAM(n, c) model as a function of c. We plot this lower bound in Figure 2. We see that the lower bound achieves a unique minimum on the interval $(0, \infty)$ and that it converges to 1 as c goes to infinity. By numerically minimizing the lower bound we obtain that the minimum of this curve is achieved at $c \approx 0.667766$ and the lower bound is ≈ 0.715071. Thus, we have the following corollary:

▶ **Corollary 15.** *For all regimes of c we have*

$$\rho(\textsc{Greedy}, \text{RTPAM}(n, c)) \geq 0.715.$$

The shape of the lower bound graph in Figure 2 is a bit strange and it suggests that our lower bound might not be tight. Therefore, we conjecture that it should be possible to strengthen the upper bound on the size of a maximum matching in RTPAM(n, c).

It is also interesting to compare the performance of Greedy on RTPAM(n, c) inputs with its performance on $G_{n,n,c/n}$ inputs. As stated in the introduction, we expect Greedy to perform worse in the RTPAM(n, c) model, because the RTPAM(n, c) model has "less randomness" in the sense of introducing correlations between consecutive online nodes that are not present in the $G_{n,n,c/n}$ model. Indeed, this intuition turns out to be correct. We plot the performance of Greedy in two models in Figure 3 and observe that Greedy on $G_{n,n,c/n}$ inputs outperforms RTPAM(n, c) for constant c.

4 Conclusion

We have introduced a new stochastic model for the online bipartite matching problem. In our model, a random Erdős-Rényi type graph is generated first. Then an input instance graph is generated in rounds where in the i^{th} round, the corresponding input type node appears consecutively n_i times where n_i is distributed according to the Poisson distribution with parameter 1.

More generally, this model is just a specific case of a broad class of stochastic online models for graph problems where type graphs are generated by some random or adversarial processes and then the i^{th} input type node occurs consecutively n_i times where n_i is determined by another random process so as to model some limited form of "locality of reference". More generally, we could use a Markov process to generate the next type node instance.

The $G_{n,n,p}$ Erdős-Rényi graphs (where $n_i = 1$ for all i) and i.i.d. models where the type graph G is determined adversarially or according to a random process (and where the i^{th} round is drawn i.i.d. from the type graph) fit within this general class of stochastic models.

As in Mastin and Jaillet [1] and Besser and Poloczek [3], we analyze the performance of the simplest greedy algorithm. As in other such studies, it is often the case that simple greedy or "greedy-like" algorithms perform well on real benchmarks or stochastic settings, well beyond what worst case analysis might suggest. Our specific RTPAM model introduces dependencies between online nodes that do not appear in other stochastic models for maximum bipartite matching. These dependencies in the RTPAM model result in some technical challenges in addition to the non-trivial analysis in Mastin and Jaillet. As in Mastin and Jaillet, our analysis falls into three classes dependening on the edge probabilities $p = c/n$. As in Mastin and Jaillet, the regimes $c = o(1)$ and $c = \omega(1)$ result in approximation ratios that approach 1 as n increases. And as in Mastin and Jaillet, we obtain an almost precise approximation ratio (modulo the estimate of the expected size of optimal matching) for the regime of constant c. Given the input dependencies our worst case bound (i.e. for the c^* that minimizes the approximation ratio) is significantly less than in Mastin and Jaillet.

As we have suggested, our RTPAM model is just a specific case of a wide class of online stochastic models that have not been studied with respect to any algorithm. We believe that such a study will be both technically interesting as well as becoming more applicable to many "real-world" settings where there is "locality of reference". Finally, we have begun an experimental study of the performance of Greedy in comparison to algorithms that exploit the underlying type graph in a distributional model (e.g., [6, 17, 2, 9, 5]).

References

1. P. Jaillet A. Mastin. Greedy online bipartite matching on random graphs, 2013. URL: https://arxiv.org/abs/1307.2536v1.
2. M. Zadimoghaddam B. Haeupler, V.S. Mirrokni. Online stochastic weighted matching: Improved approximation algorithms, WINE 2011.
3. Bert Besser and Matthias Poloczek. Greedy matching: Guarantees and limitations. *Algorithmica*, 77(1):201–234, 2017.
4. Bela Bollobás and Graham Brightwell. The width of random graph orders. *The Mathematical Scientist*, 20, 01 1995.
5. Allan Borodin, Christodoulos Karavasilis, and Denis Pankratov. An experimental study of algorithms for online bipartite matching, Unpublished work in progress, 2018.
6. Jon Feldman, Aranyak Mehta, Vahab Mirrokni, and S. Muthukrishnan. Online stochastic matching: Beating 1-1/e. In *Proc. of FOCS*, pages 117–126, 2009.
7. Gagan Goel and Aranyak Mehta. Online budgeted matching in random input models with applications to adwords. In *Proc. of SODA*, pages 982–991, 2008.
8. Carl W. Helstrom. *Statistical Theory of Signal Detection (Second Edition, Revised and Enlarged)*. International Series of Monographs in Electronics and Instrumentation. Pergamon, second edition, revised and enlarged edition, 1968.
9. Patrick Jaillet and Xin Lu. Online stochastic matching: New algorithms with better bounds. *Mathematics of Operations Research*, 39(3):624–646, 2014.
10. R. M. Karp, U. V. Vazirani, and V. V. Vazirani. An optimal algorithm for on-line bipartite matching. In *Proc. of STOC*, pages 352–358, 1990.
11. Jon M. Kleinberg. Bursty and hierarchical structure in streams. *Data Min. Knowl. Discov.*, 7(4):373–397, 2003.
12. Thomas G. Kurtz. Solutions of ordinary differential equations as limits of pure jump markov processes. *Journal of Applied Probability*, 7(1):49–58, 1970.
13. A. Mehta. Online matching and ad allocation, Theoretical Computer Science, 8(4):265–368, 2012.
14. Gordon Simons and N. L. Johnson. On the convergence of binomial to poisson distributions. *Ann. Math. Statist.*, 42(5):1735–1736, 10 1971. doi:10.1214/aoms/1177693172.
15. Kannikar Siriwong, Lester Lipsky, and Reda A. Ammar. Study of bursty internet traffic. In *Sixth IEEE International Symposium on Network Computing and Applications (NCA 2007), 12 - 14 July 2007, Cambridge, MA, USA*, pages 53–60, 2007.
16. Stats Stackexchange. What is the expectation of the absolute value of the skellam distribution? https://stats.stackexchange.com/questions/279220/what-is-the-expectation-of-the-absolute-value-of-the-skellam-distribution. Accessed: 2018-04-11.
17. A. Saberi V. H. Manshadi, S. Oveis Gharan. Online stochastic matching: Online actions based on offline statistics, Mathematics of Operations Research, 37(4):559-573, 2012.
18. Nicholas C. Wormald. Differential equations for random processes and random graphs. *Ann. Appl. Probab.*, 5(4):1217–1235, 11 1995.
19. Nicholas C. Wormald. The differential equation method for random graph processes and greedy algorithms. In *Lectures on approximation and randomized algorithms*, pages 73–155, 1999.

A Two Technical Lemmas

In this appendix, we prove two lemmas that are used in Section 3. The first lemma gives a closed-form expression for $h(x) := \mathbb{E}(\min(\mathrm{Poi}(x), \mathrm{Poi}(1)))$ in terms of the modified Bessel functions of the first kind and the Marcum's Q functions. The derivation relies on the answer from Stats Stackexchange [16].

▶ **Lemma 16** (Lemma 9 restated,[16]). *For $x > 0$ we have*

$$h(x) = \frac{1 + x - 2e^{-x-1}\left(I_0(2\sqrt{x}) + \sqrt{x}I_1(2\sqrt{x})\right) - (1-x)\left(1 - 2Q_1(\sqrt{2x}, \sqrt{2})\right)}{2}.$$

Proof. A random variable that is equal to the difference between two Poisson random variables has Skellam distribution. As the first step, we reduce the computation of $h(x)$ to the computation of the expectation of an absolute value of a Skellam distributed random variable:

$$\begin{aligned} h(x) = \mathbb{E}(\min(\mathrm{Poi}(x), \mathrm{Poi}(1))) &= \frac{\mathbb{E}(\mathrm{Poi}(x)) + \mathbb{E}(\mathrm{Poi}(1)) - \mathbb{E}(|\mathrm{Poi}(x) - \mathrm{Poi}(1)|)}{2} \\ &= \frac{1 + x - \mathbb{E}(|\mathrm{Poi}(x) - \mathrm{Poi}(1)|)}{2}. \end{aligned}$$

In the rest of the proof, we show how to compute $\mathbb{E}(|\mathrm{Poi}(x) - \mathrm{Poi}(1)|)$. From the PMF of a Skellam variable, one easily obtains the PMF of the absolute value of our Skellam variable:

$$\Pr(|\mathrm{Poi}(x) - \mathrm{Poi}(1)| = k) = \begin{cases} e^{-1-x}\left(x^{k/2}I_k(2\sqrt{x}) + x^{-k/2}I_{-k}(2\sqrt{x})\right) & \text{if } k > 0, \\ e^{-1-x}I_0(2\sqrt{x}) & \text{if } k = 0. \end{cases}$$

Then we write down the MGF and simplify it using the Marcum's Q function to get:

$$\begin{aligned} M_{|\mathrm{Poi}(x)-\mathrm{Poi}(1)|}(t) =& e^{-1-x}\Big(Q_1(\sqrt{2\exp(-t)}, \sqrt{2x\exp(t)})\exp(xe^t + e^{-t}) \\ &+ Q_1(\sqrt{2x\exp(-t)}, \sqrt{2\exp(t)})\exp(e^t + xe^{-t}) - I_0(2\sqrt{x})\Big). \end{aligned}$$

Now, we can take the derivative of the MGF at $t = 0$ to derive:

$$\mathbb{E}(|\mathrm{Poi}(x) - \mathrm{Poi}(1)|) = 2e^{-x-1}\left(I_0(2\sqrt{x}) + \sqrt{x}I_1(2\sqrt{x})\right) + (1-x)\left(1 - 2Q_1(\sqrt{2x}, \sqrt{2})\right).$$

◀

In the next lemma we prove that the function defining the differential equation in Section 3 is Lipschitz. This is needed to establish the main result of the paper via Wormald's theorem.

▶ **Lemma 17** (Lemma 11 restated). *The function $f_c(x)$ is Lipschitz on $[0, 1]$.*

Proof. We prove the statement by showing that the derivative of $f_c(x)$ is bounded on $[0, 1]$. By definition of f_c we have $f_c(x) = h(c(1-x))$. Thus, $f_c'(x) = -ch'(c(1-x))$. Hence bounding $|f_c'(x)|$ on $[0, 1]$ amounts to bounding $h'(x)$ on $[0, c]$. We compute the derivative of h as follows:

$$\begin{aligned} 2h'(x) &= 2 + 2e^{-1-x}(I_0(2\sqrt{x}) + \sqrt{x}I_1(2\sqrt{x})) - e^{-1-x}(3I_1(2\sqrt{x})/\sqrt{x} + I_0(2\sqrt{x}) + I_2(2\sqrt{x})) \\ &\quad - 2Q_1(\sqrt{2x}, \sqrt{2}) + 2(1-x)(Q_2(\sqrt{2x}, \sqrt{2}) - Q_1(\sqrt{2x}, \sqrt{2})) \\ &= 2 + 2e^{-1-x}(I_0(2\sqrt{x}) + \sqrt{x}I_1(2\sqrt{x})) - e^{-1-x}(3I_1(2\sqrt{x})/\sqrt{x} + I_0(2\sqrt{x}) + I_2(2\sqrt{x})) \\ &\quad - 2Q_1(\sqrt{2x}, \sqrt{2}) + 2(1-x)e^{-1-x}I_1(2\sqrt{x})/\sqrt{x} \\ &= e^{-1-x}(I_0(2\sqrt{x}) - I_1(2\sqrt{x})/\sqrt{x} - I_2(2\sqrt{x})) + 2 - 2Q_1(\sqrt{2x}, \sqrt{2}), \end{aligned}$$

where the second equation follows from the definition of the Marcum's Q function and the third equation follows by collecting and simplifying terms with the factor of e^{-1-x} together. We complete the proof of the lemma by bounding each of the terms.

By the definition of the Bessel functions of the first kind we have

$$\begin{aligned} I_0(2\sqrt{x}) - I_1(2\sqrt{x})/\sqrt{x} - I_2(2\sqrt{x}) &= \sum_{i=0}^{\infty} \frac{1}{i!i!} x^i - \sum_{i=0}^{\infty} \frac{1}{i!(i+1)!} x^i - \sum_{i=1}^{\infty} \frac{1}{(i-1)!(i+1)!} x^i \\ &= 1 - 1 + \sum_{i=1}^{\infty} \left(\frac{1}{i!i!} - \frac{1}{i!(i+1)!} - \frac{1}{(i-1)!(i+1)!} \right) x^i \\ &= \sum_{i=1}^{\infty} \frac{1}{i!(i+1)!} x^i \leq \sum_{i=0}^{\infty} \frac{1}{i!} x^i = e^x. \end{aligned}$$

Thus, the first term is bounded by e^c. The second term $2 - 2Q_1(\sqrt{2x}, \sqrt{2})$ is bounded by 2. This follows from interpretation of the Marcum's Q function as a probability – in particular, we have $Q_1(\sqrt{2x}, \sqrt{2}) \in [0, 1]$ (see [8]). All in all, we have $|f_c'(x)| = O(1)$ for $x \in [0, 1]$. ◀

B Figures

In this appendix, we collect all figures mentioned in the paper.

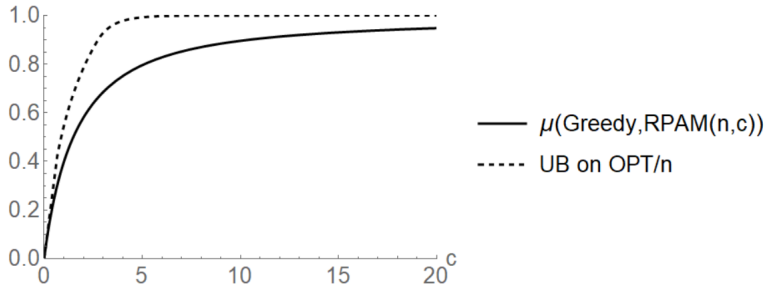


Figure 1 The exact asymptotic ratio of Greedy to n from Theorem 12, and the upper bound on the asymptotic ratio of a size of maximum matching to n from Theorem 13. Both results are plotted as functions of c in the RTPAM(n, c) model.

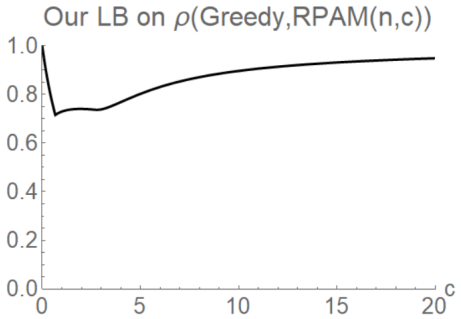


Figure 2 The lower bound on the competitive ratio for Greedy in RTPAM(n, c) model as a function of c.

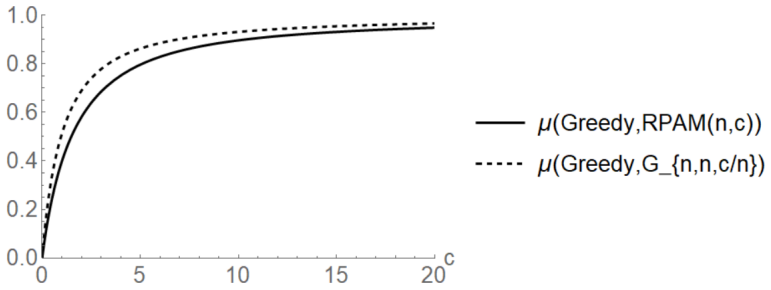


Figure 3 The exact asymptotic ratio of GREEDY to n in RTPAM(n, c) input model (Theorem 12) versus the exact asymptotic ratio of GREEDY to n in $G_{n,n,c/n}$ input model (Theorem 5).

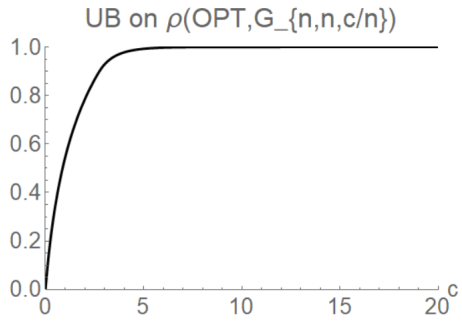


Figure 4 The upper bound on the asymptotic ratio of the size of a maximum matching in $G_{n,n,c/n}$ to n given in Theorem 4.

Semi-Direct Sum Theorem and Nearest Neighbor under ℓ_∞

Mark Braverman[1]
Department of Computer Science, Princeton University, 35 Olden St. Princeton NJ 08540, USA
mbraverm@cs.princeton.edu

Young Kun Ko
Department of Computer Science, Princeton University, 35 Olden St. Princeton NJ 08540, USA
yko@cs.princeton.edu

Abstract

We introduce semi-direct sum theorem as a framework for proving asymmetric communication lower bounds for the functions of the form $\bigvee_{i=1}^{n} f(x, y_i)$. Utilizing tools developed in proving direct sum theorem for information complexity, we show that if the function is of the form $\bigvee_{i=1}^{n} f(x, y_i)$ where Alice is given x and Bob is given y_i's, it suffices to prove a lower bound for a single $f(x, y_i)$. This opens a new avenue of attack other than the conventional combinatorial technique (i.e. "richness lemma" from [12]) for proving randomized lower bounds for asymmetric communication for functions of such form.

As the main technical result and an application of semi-direct sum framework, we prove an information lower bound on c-approximate Nearest Neighbor (ANN) under ℓ_∞ which implies that the algorithm of [9] for c-approximate Nearest Neighbor under ℓ_∞ is optimal even under randomization for both decision tree and cell probe data structure model (under certain parameter assumption for the latter). In particular, this shows that randomization cannot improve [9] under decision tree model. Previously only a deterministic lower bound was known by [1] and randomized lower bound for cell probe model by [10]. We suspect further applications of our framework in exhibiting randomized asymmetric communication lower bounds for big data applications.

2012 ACM Subject Classification Theory of computation → Communication complexity

Keywords and phrases Asymmetric Communication Lower Bound, Data Structure Lower Bound, Nearest Neighbor Search

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.6

1 Introduction

Direct Sum Theorem in communication and information complexity is a key technique in lower bounding the communication and information complexity of computing functions of the following form

$$F(\vec{x}, \vec{y}) = \bigvee_{i=1}^{n} f(x_i, y_i)$$

where Alice is given a length n string $\vec{x}$ and Bob is given another length n string $\vec{y}$ according to some distribution. Many fundamental functions, namely Disjointness and Equality (under

[1] Research supported in part by an NSF Awards, DMS-1128155, CCF- 1525342, and CCF-1149888, a Packard Fellowship in Science and Engineering, and the Simons Collaboration on Algorithms and Geometry. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

de Morgan's Law), are of the above form. Direct Sum Theorem allows us to reduce computing f to computing F. This implies that the lower bound for f can be translated to a lower bound for F. Since f is a function over a smaller number of bits (1-bit AND in the case of Disjointness) providing lower bounds for f is much easier than for F from the technical perspective.

Asymmetric communication complexity addresses a different setting where the bit-size of Bob's input is much larger than Alice's input. One example of such setting is determining whether $S \cap T = \emptyset$ when $|S| \ll |T|$ (Lopsided Disjointness). In this setting, it is more meaningful to lower bound the length of message sent by Alice and Bob separately, instead of bounding the total length of the transcript as in the symmetric setting. In asymmetric setting, the trivial protocol where Alice sends her whole input is usually the most efficient protocol in terms of total number of bits communicated between Alice and Bob.

Lower bounds in asymmetric communication are not only interesting from pure communication complexity theory perspective but also from its applications to lower bounds in data structures as seen in [11, 12, 14]. It is now a well-taught fact in graduate communication complexity classes that asymmetric communication lower bounds translate to data structure (cell probe and decision tree models) lower bounds. We explain this connection formally in the appendix Section B.

The key technical tool that was used for showing asymmetric communication lower bounds is **"Richness Lemma"** from [12]. This lemma is an extension of monochromatic rectangle lower bound for symmetric communication complexity. The main observation is that lower bounds on the height and width of any monochromatic rectangle translates to lower bounds on asymmetric communication complexity. But similar to symmetric communication setting, rectangle lower bounds usually require complicated combinatorial lemmas. Further adding to this technical complication, it is also not a great tool for analyzing the performance of randomized protocols. For randomized protocols, rectangles are no longer monochromatic but are allowed to be "roughly" monochromatic, which makes it harder to argue that such rectangle does not exist.

To avoid technical complications from asymmetric communication complexity, [13, 10] introduces a notion of "robust expansion," to lower bound the number of cells required in cell-probe data structure model even under randomization. However, this method does not imply lower bound in decision tree model as shown in [1] for the deterministic case.

Instead of using "Richness Lemma," we take an information theoretic approach in establishing asymmetric communication lower bound, similar to the one observed in [8] for lopsided disjointness. But we emphasize that unlike in [8] or lopsided disjointness, the functions in consideration are of the form

$$F(x, \vec{y}) = \bigvee_{i=1}^{n} f(x, y_i).$$

These functions are especially interesting for big data applications. In particular, it provides a lower bound for the following set of queries:

> Alice (user) with input x queries Bob (database) with $y_1, \ldots, y_n$ whether there exists a data point that satisfies certain condition (i.e. $f(x, y_i)$).

More importantly, unlike in [8], Alice's input is recycled over the singleton function f, while Bob's inputs are all distinct. Therefore, simple application of Direct Sum Theorem does not suffice for this application.

As in Direct Sum Theorem, we reduce the task of computing f to computing F. Indeed, as in symmetric communication setting, direct sum does not necessarily hold if one considers the

total length of the transcript that is the number of communicated bits. Instead, we introduce "asymmetric information cost," and analogous "asymmetric information complexity," as introduced in [8, 15] as the key complexity measure in asymmetric communication. We then show that asymmetric information cost lower bounds asymmetric communication cost with an analogous argument from symmetric communication setting.

1.1 Technical Results

Semi-Direct Sum

With a properly defined notion on asymmetric information cost, it is rather straightforward to prove the following semi-direct sum theorem.

▶ **Theorem 1** (Semi-Direct Sum (informal)). *Consider a protocol* Π *for computing* F *with the following guarantee :*

- *For all* $(x, \vec{y}) \in F^{-1}(1)$, $\Pr_\pi[\pi(x, \vec{y}) \neq 1] < \varepsilon_1$
- *For all* $(x, \vec{y}) \in F^{-1}(0)$, $\Pr_\pi[\pi(x, \vec{y}) \neq 0] < \varepsilon_0$

that is with a prior free error guarantee. Suppose μ *is a distribution such that* $X, Y_1, \ldots, Y_n$ *are i.i.d. Furthermore, suppose* $f(X, Y_i) = 1$ *with probability at most* $o(1/n)$ *under* μ*. Then there exists a protocol* Π' *(with input* X *and* Y_i*) for computing a singleton* f *with the following guarantee :*

- *For all* $(x, y_i) \in f^{-1}(1)$, $\Pr_{\pi'}[\pi'(x, y_i) \neq 1] < \varepsilon_1 + 0.01$
- *For all* $(x, y_i) \in f^{-1}(0)$, $\Pr_{\pi'}[\pi'(x, y_i) \neq 0] < \varepsilon_0 + 0.01$

with $n \cdot I_{(x,y_i)\sim\mu}(\Pi'; Y_i|X) \leq I_{(x,\vec{y})\sim\mu}(\Pi; Y_1, \ldots Y_n|X)$ *and* $|\Pi'_a| \leq |\Pi_a|$*, that is the total number of bits sent by Alice.*

This allows us to translate a lower bound on a singleton function f, to a lower bound on the whole F if the distribution for measuring the information cost is i.i.d. In particular, if one shows that for any Π' with the prior free error guarantee require $I_{(x,y_i)\sim\mu}(\Pi'; Y_i|X) > \beta$, this implies that Π requires $I_{(x,\vec{y})\sim\mu}(\Pi; \vec{Y}|X) > \beta n$.

We emphasize that the error guarantee is "prior free" in a sense that the protocol must be correct (up to some error parameters) for all inputs, instead of being correct on average under some distribution. In particular, the probability of error is only over the protocol, not the input. Under the guarantee, we can convert the prior free lower bound to a lower bound for a fixed prior using minimax argument as in [3]. More formally, we have the following theorem.

▶ **Theorem 2** (Minimax Theorem). *Fix some* $0 < \alpha < 1$*. Consider a protocol* Π *for computing* F *such that for any* $(x, \vec{y})$, $\Pr_\pi[\pi(x, \vec{y}) \neq F(x, \vec{y})] < \varepsilon/\alpha$*. Then there exists a "hard" distribution* μ *on* X *and* $\vec{Y}$ *such that for any protocol* Π' *such that* $\Pr_{(x,\vec{y})\sim\mu}[\pi'(x, \vec{y}) \neq F(x, \vec{y})] < \varepsilon$*,*

$$\frac{I_{(x,\vec{y})\sim\mu}(\Pi'; \vec{Y}|X)}{1-\alpha} \geq \sup_\nu I_{(x,\vec{y})\sim\nu}(\Pi; \vec{Y}|X).$$

Intuitively, this implies that the asymmetric information cost for protocols with distributional error guarantee and prior free error guarantee are at most constant factor within each other. In particular, setting $\alpha = 1/2$, and exhibiting a lower bound on $I_{(x,\vec{y})\sim\nu}(\Pi; \vec{Y}|X)$ for a particular ν (which is a product distribution from semi-direct sum theorem), we exhibit an existence of a hard distribution μ such that any protocol that errors at most ε on average must reveal a large amount of information about $\vec{Y}$.

The main technical disadvantage of the rectangle bound is that it is extremely challenging to give any bounds on non-product distribution over Y_i's as mentioned in [1]. We avoid this complication by a standard technique in information complexity. We first provide a lower bounds on prior-free protocols. Then using minimax argument, we can argue that a hard distribution exists **without having to explicitly construct a hard distribution**.

Nearest Neighbor Lower Bound

As the main application of our approach, we revisit c-approximate Nearest Neighbor Search under ℓ_∞-norm, improving the corresponding asymmetric communication lower bound to asymmetric information lower bound. This **strengthens the deterministic asymmetric communication lower bound given in [1] to a randomized asymmetric communication lower bound**, and obtain two corollaries for decision tree model and cell-probe model respectively. (1) For cell-probe model, this **reproves and simplifies the proof of [10]**; (2) For decision tree model, this shows that **[9] is tight for decision tree model even under randomization**, substantially improving upon [1] which only proved tightness for deterministic decision tree model. Therefore, this **closes the remaining gap for c-approximate Nearest Neighbor Search under ℓ_∞-norm under (randomized) decision tree model**.

Formally, consider the partial function

$$F(x,\vec{y}) = \begin{cases} 1 & \text{if } \exists y_i \ \|x-y_i\|_\infty \le 1 \\ 0 & \text{if } \forall y_i \ \|x-y_i\|_\infty \ge c. \end{cases}$$

for $c > 3$ with $x, y_1, \ldots, y_n \in \{0, \ldots, M\}^d$ where Alice (user) is given x and Bob (database) is given $y_1, \ldots, y_n$. The goal of the database is to compute F. [9] showed an unorthodox yet efficient, deterministic data structure which achieves $O(\log_\rho \log d)$ approximation using $O(dn^\rho \operatorname{poly}\log n)$ space. Surprisingly, [1] showed the optimality of [9] under any deterministic data structure (under decision tree model) while [10] showed that this is optimal for $n^{o(1)}$-word size (randomized) cell probe data structure. Whether randomization can improve [9] under decision tree model remained an open problem for over a decade. We answer this negatively by using semi-direct sum theorem. In particular, we prove the following asymmetric information lower bound.

▶ **Theorem 3.** *Set $\rho = (\tau \log d)^{1/c}$, and $d \ge (2 \cdot \log n)^{\frac{1}{1-\tau}}$ for some constant $1 > \tau > 0$. Let Π be a protocol (with both private and public randomness) that computes F with the following guarantee*

- *For all $(x,\vec{y}) \in F^{-1}(1)$, $\Pr[\pi(x,y) \ne 1] < 0.05$*
- *For all $(x,\vec{y}) \in F^{-1}(0)$, $\Pr[\pi(x,y) \ne 0] < 0.05$*

There exists a distribution u such that for any such Π and for any sufficiently small constant $\delta > 0$, if $|\Pi_a| \le \delta\rho \log n$ i.e. the number of bits sent by Alice, then $I_b = I_{(x,\vec{y})\sim u^{\otimes n+1}}(\Pi; \vec{Y}|X) \ge n^{1-O(\delta)}$.

This is an analogous theorem to asymmetric communication lower bound provided in [1]. But we point out that the error guarantee required is prior free and this is an information lower bound which is stronger than communication lower bound. This theorem, together with standard techniques in translating asymmetric communication lower bound to decision tree lower bound, shows that [9] is optimal under randomized decision tree model with prior free error guarantee.

[1] raised an interesting technical bottleneck that the hard distribution for randomized asymmetric communication (with distributional error guarantee) must be a *non-product distribution* over the inputs. We avoid this technical bottleneck by applying the minimax

theorem, thereby exhibiting a lower bound with distributional error guarantee over the inputs without explicitly constructing a hard (non-product) distribution. Furthermore, using the standard technique for translating asymmetric communication lower bound to cell-probe lower bound, this reproves [10].

2 Preliminary

2.1 Asymmetric Information Cost

In this section, we define analogous quantities for asymmetric communication. We refer the reader to Section A for the definitions in symmetric settings. First, recall the following natural definition for defining asymmetric communication cost.

▶ **Definition 4** (Asymmetric Communication Cost)**.** Protocol Π has asymmetric communication cost of (a, b) if $|\Pi_a| := \sum_{i:\text{odd}} |\Pi_i| \leq a$ and $|\Pi_b| := \sum_{i:\text{even}} |\Pi_i| = b$.

Since we have two parameters, naive notion of lower bound on one quantity no longer applies. Here, the lower bound will be of the form "if $|\Pi_a| < a$, then $|\Pi_b| > b(a)$."

We introduce analogous definitions for information cost, as first defined in [15], extending the intuition explained in Section A.

▶ **Definition 5** (Asymmetric Information Cost)**.** Protocol Π has asymmetric information cost of $[I_a, I_b]$ under μ if $I^\mu_a(\Pi) := I_\mu(\Pi; X|Y) \leq I_a$ and $I^\mu_b(\Pi) := I_\mu(\Pi; Y|X) \leq I_b$.

Similar to information cost of a protocol being a lower bound for communication cost of a protocol, it is straightforward prove an analogous lemma for asymmetric setting.

▶ **Lemma 6** (Asymmetric Information Cost lower bounds Communication Cost)**.** *For any protocol* Π *and any distribution* μ*,* $I^\mu_a(\Pi) \leq |\Pi_a|$ *and* $I^\mu_b(\Pi) \leq |\Pi_b|$.

Proof. Suppose at round r, Alice sends a_r bits. Now we can write the information cost incurred at round r, that is how much additional information Bob gains, as $I_\mu(M_{r+1}; X|Y, \Pi_r)$, where M_{r+1} is the message sent by Alice. Now we can write

$$I_\mu(M_{r+1}; X|Y, \Pi_r) \leq H_\mu(M_{r+1}|Y, \Pi_r) \leq a_r$$

where the last inequality holds since M_{r+1} is of length a_r and thus can have entropy of at most a_r. Applying Fact 26,

$$I_\mu(\Pi; X|Y) = \sum_r I_\mu(M_{r+1}; X|Y, M_1, \ldots, M_r) \leq \sum_r a_r = |\Pi_a|$$

Similarly we also get $I_\mu(\Pi; Y|X) \leq |\Pi_b|$. ◀

Indeed, it is no longer meaningful to argue the infimum of one quantity. Instead, we impose condition on $|\Pi_a|$ or $I^\mu_a(\Pi)$ then argue about the infimum of $I^\mu_b(\Pi)$. Then similar to distributional information complexity and prior-free information complexity as defined in [3], we can define an analogous notion for asymmetric setting conditioned on $|\Pi_a| \leq a$.

▶ **Definition 7** (Distributional Asymmetric Information Complexity)**.** Distributional asymmetric information complexity for Bob of f under μ with error ε and subject to $|\Pi_a| < a$ is defined as

$$\mathsf{IC}^{<a}_\mu(f, \varepsilon) = \inf_\Pi I^\mu_b(\Pi)$$

where the infimum is taken over the set of protocols that achieve $\Pr_{(x,y)\sim\mu, \pi\sim\Pi}[\pi(x, y) \neq f(x, y)] < \varepsilon$.

▶ **Definition 8** (Prior Free Asymmetric Information Complexity)**.** Prior free asymmetric information complexity for Bob of f with error ε and $|\Pi_a| < a$ is defined as

$$\mathsf{IC}^{<a}(f,\varepsilon) = \inf_{\Pi} \max_{\mu} I_b^{\mu}(\Pi)$$

where the infimum is taken over the set of protocols that achieve $\Pr[\pi(x,y) \neq f(x,y)] < \varepsilon$ for all (x,y).

Indeed, one can also similarly define it with an upper bound on the information revealed by Alice say I_a^{μ}, rather than $|\Pi_a|$. But for our application (to data structure lower bound), the above definition suffices. Then using a standard Minimax argument, we can also that prior free asymmetric information complexity lower bounds distributional asymmetric information complexity up to some constant factor in the error parameter.

▶ **Theorem 9** (Theorem 2 rephrased)**.** *For any f, $\varepsilon \geq 0$, and $\alpha \in (0,1)$, there exists μ on (x,y) such that*

$$\mathsf{IC}^{<a}(f,\varepsilon/\alpha) \leq \frac{\mathsf{IC}_{\mu}^{<a}(f,\varepsilon)}{1-\alpha}$$

Proof. The proof follows from a standard minimax technique (by Theorem 3.5 of [3]) for translating prior free lower bound to distribution lower bound. We define the following two-player zero-sum game, where Player 1 comes up with a protocol Π for f conditioned on $|\Pi_a| < a$ (note that such set of protocols is closed under convex combination) and Player 2 comes up with a distribution μ on (x,y)'s with the following payoff :

$$P(\Pi,\mu) := (1-\alpha)\cdot\frac{I_b^{\mu}(\Pi)}{I} + \alpha\cdot\frac{\Pr_{(x,y)\sim\mu}[\pi(x,y)\neq f(x,y)]}{\varepsilon}$$

where $I := \max_{\mu} \mathsf{IC}_{\mu}^{<a}(f,\varepsilon)$. Then the rest of the argument follows from [3]. ◀

We remark that our hard distribution for distributional error guarantee therefore is not explicitly defined since the proof is non-constructive and follows from bounding $\mathsf{IC}^{<a}(f,\varepsilon/\alpha)$. Since $\mathsf{IC}^{<a}(f,\varepsilon/\alpha)$ is maximum over the distributions, it suffices to exhibit a lower bound on a particular (in our case a product distribution) distribution. But *this does not imply that μ is a product distribution as well*, since a convex combination of product distributions is not necessarily a product distribution as well.

3 Semi-Direct Sum Theorem

In this section, we prove semi-direct sum theorem for prior free information cost, that is where the protocol is guaranteed to be correct with good probability on all inputs when the underlying distribution is a product distribution.

Let $F(x,y)$ be of form $\bigvee_{i=1}^{n} f(x,y_i)$, that is OR of functions on singletons. Recall that in symmetric setting (refer to [4]) we prove direct sum by extracting a strategy for a single copy (x_i,y_i) from a protocol that solves for $(\vec{x},\vec{y})$. Similarly, we can prove semi-direct sum theorem for those set of functions, by extracting a strategy for $f(x,y_i)$ from a protocol for $F(x,\vec{y})$. More precisely, we prove the following theorem.

▶ **Theorem 10** (Semi Direct Sum Theorem)**.** *Consider a protocol Π for computing F with the following guarantee :*

- *For all $(x,\vec{y}) \in F^{-1}(1)$, $\Pr_{\pi}[\pi(x,\vec{y}) \neq 1] < \varepsilon_1$*

- *For all* $(x, \vec{y}) \in F^{-1}(0)$, $\Pr_\pi[\pi(x, \vec{y}) \neq 0] < \varepsilon_0$
- $|\Pi_a| \leq a$ *and* $I^\mu_b(\Pi) \leq b$

where μ *is a distribution such that* $Y_1, \ldots, Y_n$ *are i.i.d. with* X *distributed independently as well. Furthermore, suppose* $f(X, Y_i) = 1$ *with probability at most* $o(1/n)$ *under* μ*. Then there exists a protocol* Π' *(with input* X *and* Y_i*) for computing a singleton* f *with the following guarantee :*

- *For all* $(x, y_i) \in f^{-1}(1)$, $\Pr_{\pi'}[\pi'(x, y_i) \neq 1] < \varepsilon_1 + 0.01$
- *For all* $(x, y_i) \in f^{-1}(0)$, $\Pr_{\pi'}[\pi'(x, y_i) \neq 0] < \varepsilon_0 + 0.01$
- $|\Pi'_a| \leq a$ *and* $I^\mu_b(\Pi') \leq b/n$.

and $|\Pi'_a| \leq |\Pi_a|$*, that is the total number of bits sent by Alice.*

Before proving the theorem, we prove necessary facts in information theory.

▶ **Proposition 11.** *Suppose* $Y_1, \ldots Y_n$ *are all independent given* X*. Then*

$$I(\Pi; Y_i | X, Y_1, \ldots Y_{i-1}) \geq I(\Pi; Y_i | X)$$

Proof. Via our independence assumption, $I(Y_i; Y_1, \ldots, Y_{i-1}|X) = 0$. Then applying Fact 26, we get the desired inequality. ◀

▶ **Lemma 12** (Semi Direct Sum)**.** *Suppose given* X*,* Y_i*'s are i.i.d. Then*

$$\frac{1}{n} \cdot I(\Pi; Y|X) \geq I(\Pi; Y_i|X)$$

Proof. First we show that $I(\Pi; Y|X) \geq \sum_i I(\Pi; Y_i|X)$. By Fact 27 we have

$$I(\Pi; Y_1, \ldots, Y_n|X) = I(\Pi; Y_1|X) + I(\Pi; Y_2|Y_1, X) + \ldots + I(\Pi; Y_n|Y_1, \ldots, Y_{n-1}, X)$$

Now for each term $I(\Pi; Y_i|Y_1, \ldots, Y_{i-1}, X)$, we can lower bound it with $I(\Pi; Y_i|X)$ from Proposition 11 due to our assumption on independence. Applying the lower bound term by term we have $I(\Pi; Y|X) \geq \sum_i I(\Pi; Y_i|X)$. By our assumption on the distribution, $I(\Pi; Y_i|X) = I(\Pi; Y_j|X)$ for all $i \neq j$. Thus we have the desired inequality. ◀

Proof of Theorem 10.

Protocol 1 Protocol Π'

1. Alice and Bob jointly and publicly samples $J \in [n]$.
2. Bob privately samples $Y_1 \ldots Y_{J-1}, Y_{J+1} \ldots Y_n$.
3. Alice and Bob set $X = x$ and $Y_J = y$ then run protocol Π on $(x, \vec{y})$

Consider Protocol 1. First observe that $F(X, Y) = f(X, Y_J)$ with high probability, since by our assumption on the density of $f^{-1}(0)$ under μ,

$$\Pr\left[\bigvee_{\substack{i=1 \\ i \neq J}}^{n} f(X, Y_i) = 0\right] \leq (1 - o(1/n))^{n-1} = o(1).$$

Therefore, Π' satisfies the claimed guarantee if Π has the guarantee. Also $|\Pi'_a| = |\Pi_a|$ by design. The bound on $I^\mu_b(\Pi')$ follows from Proposition 11 and Lemma 12 via following observation

$$I(\Pi'; Y|X) = I(\Pi; Y|X) \leq I(\Pi, J; Y|X) = I(J; Y|X) + I(\Pi; Y_J|J, X)$$
$$= I(\Pi; Y_J|J, X) = \frac{1}{n} \cdot \sum_{i=1}^{n} I(\Pi; Y_i|X) \leq \frac{I(\Pi; Y_1 \ldots Y_n|X)}{n}.$$ ◀

As a contrapositive of Theorem 10, we obtain the following corollary which will be the main component in establishing asymmetric information lower bound.

▶ **Corollary 13.** *Suppose there exists a product distribution μ on X and Y with $\Pr_\mu[f(x,y) = 1] = o(1/n)$ such that for any protocol Π that computes f with $\Pr[f(x,y) \neq \pi(x,y)] < \varepsilon$ for all (x,y) with $|\Pi_a| < a$, $I_\mu(\Pi; Y|X) \geq b$. Then there exists a (product distribution) μ_n such that for any protocol Π^n that computes F with $\Pr[F(x,\vec{y}) \neq \pi_n(x,\vec{y})] < \varepsilon + 0.01$ for all $(x,\vec{y})$ with $|\Pi^n_a| < a$, $I_{\mu_n}(\Pi^n; \vec{Y}|X) \geq n \cdot b$. In other words, $\mathsf{IC}^{<a}(f,\varepsilon) \geq n \cdot b$.*

4 Nearest Neighbor in ℓ_∞ Lower Bound

In this section, we prove Theorem 3. To utilize Theorem 10, we focus on prior free information cost lower bound for the following function for $x, y \in \{0, \ldots, M\}^d$.

$$f(x,y) := \begin{cases} 1 & \text{if } \|x-y\|_\infty \leq 1 \\ 0 & \text{if } \|x-y\|_\infty \geq c \end{cases}$$

We focus on lower bounding $\mathsf{IC}^{<a}(f,\varepsilon)$ for some sufficiently small constant ε. The main idea of the proof is that any protocol that achieves the desired error guarantee must be distinguishing between $f^{-1}(1)$ and $f^{-1}(0)$ for any given x. In other words, whether y is in the neighborhood of x or not.

To make the proof simpler, we prove the following Compression Lemma whose proof we attach in Section C which allows us to assume without loss of generality (with some minor costs) that the protocol is one round where Bob's reply is a single bit.

▶ **Lemma 14** (Compression for Bob)**.** *Consider Π such that $I_{u^{\otimes 2}}(\Pi; X|Y) < I_a$ and $I_{u^{\otimes 2}}(\Pi; Y|X) < I_b$ that computes f. Further, assume that $I_a \cdot I_b = o(1)$. Then Π can be compressed to a one round protocol $\tau \sim \Pi'$ with $I_{u^{\otimes 2}}(\Pi'; X|Y) < O(I_a)$ and $I_{u^{\otimes 2}}(\Pi'; Y|X) < O\left(H(\sqrt{I_a I_b})\right)$ with the following guarantee:*

- *For $(x,y) \in f^{-1}(1)$, $\Pr[\tau(x,y) \neq 1] < \varepsilon_1 + 0.05$*
- *There exists $Z \subset f^{-1}(0)$, with $\mu(Z) = \mu(f^{-1}(0)) - 0.01$ such that for all $(x,y) \in S$, $\Pr[\tau(x,y) \neq 0] < \varepsilon_0 + 0.05$*

where $\varepsilon_0 = \max_{(x,y)\in f^{-1}(0)} \Pr[\pi(x,y) \neq f(x,y)]$ and similarly $\varepsilon_1 = \max_{(x,y)\in f^{-1}(1)} \Pr[\pi(x,y) \neq f(x,y)]$.

We also prove the following simple observation which allows us to assume without loss of generality that only Bob has access to private randomness for a single round protocol.

▶ **Lemma 15.** *Suppose there is a single round protocol Π where*

- *Alice and Bob both have access to both private and public randomness.*
- *Alice sends at most a-bits $|\Pi_a| \leq a$*
- *Bob sends at most b-bits of information $I(\Pi; Y|X) \leq b$.*

Then there is a protocol Π' where Alice does not have access to private randomness but $|\Pi'_a| \leq a$ and $I(\Pi'; Y|X) \leq b$

Proof. Consider the following simple modification to Π' from Π. Alice additionally samples from public randomness R^a_{pub} (thereby revealing the randomness of Bob) instead of using private randomness and follows Π. Bob ignores R^a_{pub} and follows Π. By design $|\Pi'_a| \leq a$. Let M_b denote the reply by Bob and M_a the message by Alice. Then

$$I(\Pi'; Y|X) = I(M_b; Y|X, R^a_{pub}, M_a) \leq I(M_b; Y|X, M_a) = I(\Pi; Y|X)$$

since $I(Y; R^a_{pub}|M_a, X) = 0$. ◀

Lemma 15 implies that it suffices to lower bound the case where Alice does not have access to private randomness. Then we set

$$u(x) := \begin{cases} 2^{-2\rho^x} & \text{if } x \in [M] \\ 1 - \sum_{i\in[M]} 2^{-2\rho^i} & \text{if } x = 0. \end{cases}$$

with $u(\vec{x})$ with $\vec{x}$ as a d-dimensional vector defined as a product of $u(x_i)$'s. Then we prove the main technical theorem whose proof is attached in Section D.

▶ **Theorem 16** (Single Function Lower Bound). *Let* Π *be a one round protocol where Alice does not have access to private randomness, Bob replies with one bit and computes* f *with the following guarantee*

- *For* $(x, y) \in f^{-1}(1)$, $\Pr[\pi(x, y) \neq 1] < 0.1$
- *There exists* $S \subset f^{-1}(0)$, *with* $u(S) \geq u(f^{-1}(0)) - 0.01$ *such that for all* $(x, y) \in S$, $\Pr[\pi(x, y) \neq 0] < 0.1$

For such Π, *for any sufficiently small constant* $\delta > 0$, *if* $I_a = I_{(x,y)\sim u^{\otimes 2}}(\Pi; X|Y) \leq \delta\rho \log n$, *then* $I_b = I_{(x,y)\sim u^{\otimes 2}}(\Pi; Y|X) \geq n^{-O(\delta)}$.

The main intuition of the proof follows from bounding ℓ_1-norm between transcripts from $(x, y) \in f^{-1}(1)$ and $(x, y) \in Z$. If the information sent by Bob is lower than the claimed bound, then the protocol cannot distinguish between $f^{-1}(1)$ and $f^{-1}(0)$ (more technically close in terms of ℓ_1-norm) which is a contradiction.

Then combining Theorem 16 with Lemma 14 and Lemma 15, we get the following prior free bound as a corollary conditioned on $|\Pi_a| \leq \delta\rho \log n$.

▶ **Corollary 17.** *Let* Π *be any protocol that computes* f *with the following guarantee*

- *For all* $(x, y) \in f^{-1}(1)$, $\Pr[\pi(x, y) \neq 1] < 0.05$
- *For all* $(x, y) \in f^{-1}(0)$, $\Pr[\pi(x, y) \neq 0] < 0.05$

For such Π, *for any sufficiently small constant* $\delta > 0$, *if* $I_a = I_{(x,y)\sim u\times u}(\Pi; X|Y) \leq \delta\rho \log n$, *then* $I_b = I_{(x,y)\sim u\times u}(\Pi; Y|X) \geq n^{-O(\delta)}$.

Proof. Suppose we have a protocol that computes f with the guarantee, with $I_a \leq \delta\rho \log n$ and $I_b = I(\Pi; Y|X) < n^{-\omega(\delta)}$. Then by Lemma 14, we can compress the protocol to a one-round protocol with 1-bit response from Bob with information cost $O(\delta\rho \log n)$ and $n^{-\omega(\delta)}$ respectively for Alice and Bob, with the following error guarantee.

- For all $(x, y) \in f^{-1}(1)$, $\Pr[\tau(x, y) \neq 1] < 0.1$
- There exists $S \subset f^{-1}(0)$, with $u(S) \geq u(f^{-1}(0)) - 0.01$ such that for all $(x, y) \in S$, $\Pr[\tau(x, y) \neq 0] < 0.1$

This is indeed a contradiction to Theorem 16. ◀

Then finally combining Corollary 17 with Corollary 13, and with a guarantee that $f(x, y) = 1$ with probability $o(1/n)$, the proof of which we append in Claim 40, we get a lower bound for any protocol that computes $F(x, \vec{y})$ with prior free error guarantee, when the information cost is measured over the distribution where all X and Y's are distributed according to u.

▶ **Theorem 18.** *Let* Π *be any protocol that computes* F *with the following guarantee*

- *For all* $(x, \vec{y}) \in F^{-1}(1)$, $\Pr[\pi(x, y) \neq 1] < 0.04$
- *For all* $(x, \vec{y}) \in F^{-1}(0)$, $\Pr[\pi(x, y) \neq 0] < 0.04$

For such Π, *for any sufficiently small constant* $\delta > 0$, *if* $|\Pi_a| \leq \delta\rho \log n$, *then* $I_b = I(\Pi; Y|X) \geq n^{1-O(\delta)}$.

Via the minimax argument, we get the following for distributional error setting (from applying Theorem 2)

▶ **Corollary 19.** *For any sufficiently small constant $\delta > 0$, there exists μ such that for any protocol Π that computes F with $\Pr_{(x,\vec{y})\sim\mu}[\pi(x,\vec{y}) \neq F(x,\vec{y})] < 0.01$, if $|\Pi_a| \leq \delta\rho \log n$, then $I_b = I(\Pi; Y|X) \geq n^{1-O(\delta)}$.*

This yields the desired data structure lower bounds from the translations in Section B. For the definition of data structures, we refer the reader to Section B as well.

▶ **Corollary 20** (Cell-Probe Lower Bound)**.** *Consider any randomized cell-probe data structure solving d-dimensional near-neighbor search under ℓ_∞ with approximation factor $c = O(\log_\rho \log d)$, with the guarantee that if there exists $y_i \in \vec{y}$ that is close to x, the querier outputs 1 with > 0.95 probability. If the word size is $w = n^{1-\delta}$ for some $\delta > 0$, the data structure requires space $n^{\Omega(\rho/t)}$ for query time t.*

▶ **Corollary 21** (Decision Tree Lower Bound)**.** *Let $\delta > 0$ be arbitrary constant. A decision tree of depth $r = n^{1-2\delta}$ and node size $w = n^\delta$ that solves d-dimensional near-neighbor search under ℓ_∞ with approximation $c = O(\log_\rho \log d)$, must have size $s = n^{\Omega(\rho)}$.*

References

1. Alexandr Andoni, Dorian Croitoru, and Mihai Patrascu. Hardness of nearest neighbor under l-infinity. In *Foundations of Computer Science, 2008. FOCS'08. IEEE 49th Annual IEEE Symposium on*, pages 424–433. IEEE, 2008.
2. Boaz Barak, Mark Braverman, Xi Chen, and Anup Rao. How to compress interactive communication. *SIAM Journal on Computing*, 42(3):1327–1363, 2013.
3. Mark Braverman. Interactive information complexity. *SIAM Journal on Computing*, 44(6):1698–1739, 2015. doi:10.1137/130938517.
4. Mark Braverman, Ankit Garg, Denis Pankratov, and Omri Weinstein. From information to exact communication. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 151–160. ACM, 2013.
5. Mark Braverman and Anup Rao. Information equals amortized communication. *IEEE Transactions on Information Theory*, 60(10):6058–6069, 2014.
6. Amit Chakrabarti, Yaoyun Shi, Anthony Wirth, and Andrew Yao. Informational complexity and the direct sum problem for simultaneous message complexity. In *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*, pages 270–278. IEEE, 2001.
7. Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012.
8. Anirban Dasgupta, Ravi Kumar, and D. Sivakumar. Sparse and lopsided set disjointness via information theory. In Anupam Gupta, Klaus Jansen, José Rolim, and Rocco Servedio, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 517–528, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.
9. Piotr Indyk. On approximate nearest neighbors under l∞ norm. *Journal of Computer and System Sciences*, 63(4):627–638, 2001.
10. Michael Kapralov and Rina Panigrahy. Nns lower bounds via metric expansion for l∞ and emd. In *International Colloquium on Automata, Languages, and Programming*, pages 545–556. Springer, 2012.
11. Peter Bro Miltersen. Lower bounds for union-split-find related problems on random access machines. In *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, pages 625–634. ACM, 1994.

12 Peter Bro Miltersen, Noam Nisan, Shmuel Safra, and Avi Wigderson. On data structures and asymmetric communication complexity. In *Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*, pages 103–111. ACM, 1995.
13 Rina Panigrahy, Kunal Talwar, and Udi Wieder. Lower bounds on near neighbor search via metric expansion. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, pages 805–814. IEEE, 2010.
14 Mihai Patrascu. *Lower Bound Techniques for Data Structures*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2008. AAI0821553.
15 Sivaramakrishnan Natarajan Ramamoorthy and Anup Rao. How to compress asymmetric communication. In *Proceedings of the 30th Conference on Computational Complexity*, pages 102–123. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015.

A Information Theory

In this section, we provide necessary backgrounds on information theory and information complexity that are used in this paper. For further reference, we refer the reader to [7] and [6, 3, 4, 2, 5].

▶ **Definition 22** (Entropy). The entropy of a random variable X is defined as

$$H(X) := \sum_x \Pr[X = x] \log \frac{1}{\Pr[X = x]}.$$

Similarly, the conditional entropy is defined as

$$H(X|Y) := \mathop{\mathbb{E}}_Y \left[\sum_x \Pr[X = x|Y = y] \log \frac{1}{\Pr[X = x|Y = y]} \right].$$

As an abuse of notation, we also denote binary entropy function $H : [0, 1] \to [0, 1]$ as

$$H(p) := p \log \frac{1}{p} + (1 - p) \log \frac{1}{1 - p}$$

▶ **Definition 23** (Mutual Information). Mutual information between X and Y (conditioned on Z) is defined as

$$I(X; Y|Z) := H(X|Z) - H(X|YZ).$$

▶ **Definition 24** (KL-Divergence). KL-Divergence between two distributions μ and ν is defined as

$$D(\mu||\nu) := \sum_x \mu(x) \log \frac{\mu(x)}{\nu(x)}.$$

In order to bound mutual information, it suffices to bound KL-divergence, due to following fact.

▶ **Fact 25** (KL-Divergence and Mutual Information). *The following equality between mutual information and KL-Divergence holds*

$$I(A; B|C) = \mathop{\mathbb{E}}_{B,C} \left[D(A|_{B=b,C=c}||A|_{C=c}) \right].$$

▶ **Fact 26.** *Let A, B, C, D be random variables such that $I(A; D|C) = 0$. Then*

$$I(A; B|C) \leq I(A; B|C, D).$$

▶ **Fact 27** (Mutual Information Chain Rule)**.**

$$I(A; B_1, \dots B_n|C) = \sum_{i=1}^{n} I(A; B_i|C, B_1, \dots B_{i-1})$$

▶ **Fact 28.** *Let $p(x, y)$ and $q(x, y)$ be joint distributions of two random variables X and Y. Then*

$$D(p(x, y)||q(x, y)) = D(p(x)||q(x)) + \mathop{\mathbb{E}}_{x\sim p} D(p(y|x)||q(y|x))$$

To provide context to asymmetric information costs that will be introduced in the next section, we first introduce usual definition of information cost of f.

▶ **Definition 29** (Information Cost)**.** The information cost of Protocol Π with input X for Alice Y for Bob under distribution μ is defined as

$$\mathsf{IC}_\mu(\Pi) = I_\mu(\Pi; X|Y) + I_\mu(\Pi; Y|X)$$

where $I_\mu(\Pi; X|Y)$ denote mutual information where X and Y are distributed according to μ, similarly for $I_\mu(\Pi; Y|X)$.

Intuitively, $I(\Pi; X|Y)$-term captures how much information Alice must inject to the transcript to compute F. Similarly $I(\Pi; Y|X)$-term captures how much information Bob must inject to the transcript. This intuition comes useful in defining analogous quantities for the asymmetric communication setting.

B Data Structure and Asymmetric Communication Lower Bound

In this section, we introduce relevant models in data structure for which an asymmetric communication lower bound translates to their lower bounds.

Cell Probe Data Structure

Cell probe data structure, which is similar to Random Access machine, can be formally defined as following.

▶ **Definition 30** (Cell-Probe)**.** Cell Probe data structure is defined as the following model for computing F. Alice (user) sends a cell-address in each round. Bob (database) answers with the contents in the queried cell. Word size w is the maximum number of bits in the cell.

The following classic "translation" theorem holds.

▶ **Theorem 31** (Lemma 1 of [12])**.** *If there exists a (randomized) cell-probe database that solves F with cell size s, word size w and query time t, then there exists $(2t \log s, 2tw)$-(randomized) communication protocol for F.*

In other words, asymmetric communication lower bounds translates to cell-probe database lower bounds. This is the key observation in [1].

Decision Tree Data Structure

Decision tree data structure model is formally defined as following.

▶ **Definition 32** (Decision Tree). Depending on $\vec{y}$ and the randomness, Bob constructs a decision tree $T_{\vec{y}}$, a complete binary tree, in which:

- Each node v contains a predicate function $f_v : \mathcal{X} \to \{0,1\}$ where $f_v \in \mathcal{F}$, the set of allowed predicates.
- Each edge is labeled 0 or 1, which denotes the answer to the parent's f_v.
- Each leaf is labeled 0 or 1, which denotes the final output to F.

Size s of the tree is the number of nodes. Depth r is the depth of the tree. Predicate size w is $\log_2 |\mathcal{F}|$.

Under this model, a more efficient translation theorem holds which results in better lower bound under decision tree model.

▶ **Theorem 33** ([1]). *If there exists a (randomized) decision tree that solves F with size s, depth d, and node size w, then there exists $(O(\log s), O(dw \log s))$-(randomized) communication protocol for F.*

As a corollary of Lemma 14 and Theorem 3, we get the following corollaries in data structure lower bound. Combining Theorem 31 with Theorem 18 (or Corollary 19 depending on the error guarantee), we get the following corollary.

▶ **Corollary 20.** *Consider any randomized cell-probe data structure solving d-dimensional near-neighbor search under ℓ_∞ with approximation factor $c = O(\log_\rho \log d)$, with the guarantee that if there exists $y_i \in \vec{y}$ that is close to x, the querier outputs 1 with > 0.95 probability. If the word size is $w = n^{1-\delta}$ for some $\delta > 0$, the data structure requires space $n^{\Omega(\rho/t)}$ for query time t.*

Combining Theorem 33 with Theorem 18 (or Corollary 19 depending on the error guarantee), we get the following corollary.

▶ **Corollary 21.** *Let $\delta > 0$ be arbitrary constant. A decision tree of depth $r = n^{1-2\delta}$ and node size $w = n^\delta$ that solves d-dimensional near-neighbor search under ℓ_∞ with approximation $c = O(\log_\rho \log d)$, must have size $s = n^{\Omega(\rho)}$.*

C Proof of Compression Lemma

In this section, we show that any protocol such that Alice sends at most I_a bits, and Bob sends at most I_b bits of information with $I_b \ll 1$ and $I_a \cdot I_b = o(1)$ can be compressed to a one-round protocol such that Alice sends at most $O(I_a)$ bits and Bob sends at most 1 bits and $H(\sqrt{I_a I_b})$-bits of information.

First we use the following compression theorem to compress the size of the transcript where τ is the resulting compressed protocol:

▶ **Theorem 34** (Theorem 1.2 in [15]). *Suppose $I_a = \omega(1)$. Then any protocol Π such that Alice sends at most I_a-bits of information and Bob sends at most I_b-bits of information can be simulated with $O(I_a 2^{O(I_b)})$ bits of communication with the following guarantee :*

- *For $(x,y) \in F^{-1}(1)$, $\Pr[\pi(x,y) \neq \tau(x,y)] < \varepsilon_1 + 0.01$*
- *There exists $\Gamma \subset F^{-1}(0)$, with $\mu(\Gamma) = \mu(F^{-1}(0)) - 0.01$ such that for all $(x,y) \in \Gamma$, $\Pr[\pi(x,y) \neq \tau(x,y)] < \varepsilon_0 + 0.01$*

where ε_1 and ε_0 refers to the respective guarantee for the original protocol Π.

▶ Remark. Theorem 1.2 in [15] is not stated as the above form. But setting the output as 1 on high divergence (x, y) pairs and adjusting the constants give the above guarantee. Also note that $2^{O(I_b)} \leq 2$ in our setting where $I_b \ll 1$.

Theorem 34 allows us to assume the length of the protocol $|\Pi| = |\Pi_a| + |\Pi_b| \leq O(I_a)$ for our setting. Now we introduce the following round compression protocol and its guarantee as Lemma 14.

Protocol 2 Compression Protocol

- Alice samples $\pi \sim \Pi_x$, the distribution of protocol conditioned on Alice's input x.
- Alice sends π to Bob
- Bob answers 1 if π agrees with his input y and private randomness r_b. Rejects otherwise.

▶ **Lemma 14.** *Consider* Π *such that* $I(\Pi; X|Y) < I_a$ *and* $I(\Pi; Y|X) < I_b$ *that computes* f *under the distribution* μ*. Further, assume that* $I_a \cdot I_b = o(1)$*. Then* Π *can be compressed to a one round protocol* $\tau \sim \Pi'$ *with* $I(\Pi'; X|Y) < O(I_a)$ *and* $I(\Pi'; Y|X) < O\left(H(\sqrt{I_a I_b})\right)$ *with the following guarantee:*

- *For* $(x, y) \in f^{-1}(1)$, $\Pr[\tau(x, y) \neq 1] < \varepsilon_1 + 0.05$
- *There exists* $Z \subset f^{-1}(0)$*, with* $\mu(Z) = \mu(f^{-1}(0)) - 0.01$ *such that for all* $(x, y) \in S$, $\Pr[\tau(x, y) \neq 0] < \varepsilon_0 + 0.05$

where $\varepsilon_0 = \max_{(x,y)\in f^{-1}(0)} \Pr[\pi(x, y) \neq f(x, y)]$ *and similarly* $\varepsilon_1 = \max_{(x,y)\in f^{-1}(1)} \Pr[\pi(x, y) \neq f(x, y)]$.

Towards proving Lemma 14, we first prove two facts on non-negative numbers.

▶ **Claim 35.** *Let* $\Pi_{x,y}$ *denote the distribution of transcript conditioned on* $X = x$ *and* $Y = y$*. Similarly let* Π_x *denote the distribution conditioned on* $X = x$*. Similarly let* $M_i|_{M_{<i},x,y}$ *and* $M_i|_{M_{<i},x}$ *denote the distribution of message at* i*-th round conditioned on all previous messages and* x, y *or just* x*. Then*

$$D(\Pi_{x,y}||\Pi_x) = \sum_{i=1}^{R} \mathop{\mathbb{E}}_{M_{<i}|x,y} D(M_i|_{M_{<i},x,y}||M_i|_{M_{<i},x})$$

Proof. Follows from Fact 28. ◀

▶ **Claim 36.** *Consider* R *non-negative numbers* $I_1, \ldots I_R$*. Then*

$$\sum_{r=1}^{R} \sqrt{I_r} \leq \sqrt{R \cdot \sum_{r=1}^{R} I_r}$$

Proof. Follows from Cauchy-Schwarz. ◀

Proof of Lemma 14. Note that without loss of generality, the total amount of bits communicated in Protocol 2 is $|\Pi| + 1 = O(I_a)$ since one can compress the transcript using Theorem 34. This immediately implies that $I(\Pi'; X|Y) < O(I_a)$, since the number of bits sent by Alice is at most $O(I_a)$. Similarly, it also implies that the number of round (R) is at most $O(I_a)$.

Now, to prove the full lemma, it suffices to bound the probability of Bob sending 0. We bound by computing the probability of Bob rejecting the transcript at each round $i + 1$, (i.e. the odd round messages) then summing up these probabilities.

Let M_i denote the message sent in i-th round of the protocol. If $i+1$ is odd (the round where Bob sends message), then we have

$$M_{i+1}|M_{\leq i}, X, Y = M_{i+1}|M_{\leq i}, Y$$

since it is a protocol and Bob's response only depends on Y and $M_{\leq i}$. Then by Pinsker's inequality, we bound the error of sampling. The probability of making error at $i+1$-th round can be bounded using Pinsker's inequality as

$$\begin{aligned}
&\mathop{\mathbb{E}}_{M_{\leq i},X,Y}[\|M_{i+1}|M_{\leq i}, X, Y - M_{i+1}|M_{\leq i}, X\|_1] \\
&\leq O\left(\mathop{\mathbb{E}}_{M_{\leq i},X,Y}\left[\sqrt{D(M_{i+1}|M_{\leq i}, X, Y||M_{i+1}|M_{\leq i}, X)}\right]\right) \\
&\leq O\left(\sqrt{\mathop{\mathbb{E}}_{M_{\leq i},X,Y}[D(M_{i+1}|M_{\leq i}, X, Y||M_{i+1}|M_{\leq i}, X)]}\right) \qquad (1)
\end{aligned}$$

where the last inequality is using the concavity of $\sqrt{x}$. Then we can sum and bound the probability of error as

$$\begin{aligned}
&\sum_{\substack{i=0\\ i:odd}}^{R-1} \mathop{\mathbb{E}}_{M_{\leq i},X,Y}[\|M_{i+1}|M_{\leq i}, X, Y - M_{i+1}|M_{\leq i}, X\|_1] \\
&\leq O\left(\sum_{\substack{i=0\\ i:odd}}^{R-1} \sqrt{\mathop{\mathbb{E}}_{M_{\leq i},X,Y}[D(M_{i+1}|M_{\leq i}, X, Y||M_{i+1}|M_{\leq i}, X)]}\right) \\
&\leq O\left(\sqrt{R \cdot \sum_{\substack{i=0\\ i:odd}}^{R-1} \mathop{\mathbb{E}}_{M_{\leq i},X,Y}[D(M_{i+1}|M_{\leq i}, X, Y||M_{i+1}|M_{\leq i}, X)]}\right) \\
&\leq O\left(\sqrt{R \cdot \mathop{\mathbb{E}}_{X,Y}[D(\Pi_{X,Y}||\Pi_X)]}\right) = O\left(\sqrt{R \cdot I_b}\right) \qquad (2)
\end{aligned}$$

where the second inequality follows from Claim 36 and the third inequality follows from Claim 35. Now $R < O(I_a)$ from the total number of bits communicated by Alice, which upper bounds the number of rounds, is $O(I_a)$ from Theorem 34. This ℓ_1-norm bound implies that Bob answers 0 with at most $O(\sqrt{I_a I_b})$-probability in expectation over (x, y). Then the amount of information that Alice learns is at most $O\left(H(\sqrt{I_a I_b})\right)$.

It remains to show that the error rate guarantee is preserved. We divide into two cases.

- If $f(x, y) = 1$, by design there is no guarantee lost on $(x, y) \in f^{-1}(1)$, except the loss from Theorem 34.
- Now suppose $f(x, y) = 0$. Recall that the compression fails with probability at most $O(\sqrt{I_a I_b})$ in expectation. Let $Z^0 := \{(x, y)|\|\Pi|X = x, Y = y - \Pi|X = x\|_1 < (I_a I_b)^{2/3}\}$. By Markov's inequality, $\mu(Z^0) \geq 1 - o(1)$. Set $Z = Z^0 \cap \Gamma \cap f^{-1}(0)$, where Γ is the set from Theorem 34. It is easy to check that the probability of error only increases by $o(1)$ for any $(x, y) \in Z$ since $(I_a I_b)^{2/3} = o(1)$ by assumption while being in Γ only increases the error rate by 0.01. Thus Z indeed satisfies the guarantee. ◀

D Omitted Proof from Section 4

Recall that

$$u(x) := \begin{cases} 2^{-2\rho^x} & \text{if } x \in [M] \\ 1 - \sum_{i\in[M]} 2^{-2\rho^i} & \text{if } x = 0. \end{cases}$$

with $u(x)$ defined as the product of $u(x_i)$'s. Under such u, we make use of the following lemma from [1].

▶ **Lemma 37** (Isoperimetric Inequality (Lemma 9 of [1])). *Consider any set $S \subseteq \{0,\dots M\}^d$. Let $N(S)$ be the set of points at distance at most 1 from S under ℓ_∞. Then $u(N(S)) \geq u(S)^{1/\rho}$.*

Then we get the following corollary in terms of KL-Divergence.

▶ **Corollary 38.** *Let $u|_S$ denote u restricted on a subset $S \subseteq \{0,\dots M\}^d$. Then for any S*

$$u(N(S)) \geq 2^{-D(u|_S||u)/\rho}$$

Proof. Observe that $D(u|_S||u) = -\log u(S)$ or $u(S) = 2^{-D(u|_S||u)}$. Then Lemma 37 implies the desired inequality. ◀

We also state the following simple fact about norm.

▶ **Fact 39** (ℓ_1-norm convex). *Consider a family of distributions μ and $\{\nu_i\}_i$. Then*

$$\|\mu - \mathop{\mathbb{E}}_i[\nu_i]\|_1 \leq \mathop{\mathbb{E}}_i[\|\mu - \nu_i\|_1]$$

Proof. Follows from the convexity of ℓ_1-norm. ◀

Now we are ready to prove Theorem 16.

Proof of Theorem 16. Let $a := \delta\rho\log n$ and $b := 2^{-2000a/\rho}$. Recall that we assume that the protocol is one round, therefore the distribution on the protocol Π is on Alice's message π_a and Bob's message π_b. Also let $\Pi_b|\pi_a$ be the distribution on Bob's message given Alice's message as π_a. If $\pi_b \sim \Pi_b|\pi_a$, note that each Bob's message π_b conditioned on π_a induces a prior on Bob's input y, μ_{π_b}. Suppose further $\mathbb{E}_{\pi_b\sim\Pi_b|\pi_a} D(\mu_{\pi_b}||u) < 10b$. Note that such message π_a must exist by Markov argument.

Since Alice does not have access to private randomness, prior on X conditioned on π_a is exactly a subset on $\{0,\dots M\}^d$, which we denote as S_{π_a}.

With π_a fixed, with an abuse of notation let $\Pi_{x,y}$ denote the distribution on the transcript (the remaining message, Π_b) conditioned on input being (x,y) and Alice's message. And let μ denote the distribution on x conditioned on the message π_a, that is $u|_{S_{\pi_a}}$. Note that Π_{x,y_1} and Π_{x,y_2} are close in ℓ_1 norm if $f(x,y_1) = f(x,y_2) = 1$ or $(x,y_1),(x,y_2) \in Z$, since the protocol is a one round protocol and Alice's message is fixed. In particular,

$$\|\Pi_{x,y_1} - \Pi_{x,y_2}\|_1 < 0.2$$

since both Π_{x,y_1} and Π_{x,y_2} outputs $f(x,y_1) = f(x,y_2)$ with at least 0.9 probability for $f^{-1}(1)$ and Z. Similarly, if $(x,y_1) \in Z$ and $f(x,y_2) = 1$, $\|\Pi_{x,y_1} - \Pi_{x,y_2}\|_1 > 1.8$. Now consider

$$\nu_{f^{-1}(0)} := \mathop{\mathbb{E}}_{x\sim\mu} \mathop{\mathbb{E}}_{\substack{y\sim u \\ (x,y)\in Z}} [\Pi_{x,y}]$$

$$\nu_{f^{-1}(1)} := \mathop{\mathbb{E}}_{x\sim\mu} \mathop{\mathbb{E}}_{\substack{y\sim u \\ f(x,y)=1}} [\Pi_{x,y}]$$

First observe that from Fact 39 if $(x, y) \in Z$,

$$\|\Pi_{x,y} - \nu_{f^{-1}(0)}\|_1 \leq \mathop{\mathbb{E}}_{x\sim\mu} \mathop{\mathbb{E}}_{\substack{y'\sim u \\ (x,y')\in Z}} [\|\Pi_{x,y} - \Pi_{x,y'}\|_1] < 0.2 \tag{3}$$

Similarly, if $f(x, y) = 1$,

$$\|\Pi_{x,y} - \nu_{f^{-1}(1)}\|_1 \leq \mathop{\mathbb{E}}_{x\sim\mu} \mathop{\mathbb{E}}_{\substack{y'\sim u \\ f(x,y')=1}} [\|\Pi_{x,y} - \Pi_{x,y'}\|_1] < 0.2. \tag{4}$$

Then by triangular inequality if $(x, y_1) \in Z$ and $f(x, y_2) = 1$,

$$1.8 < \|\Pi_{x,y_1} - \Pi_{x,y_2}\|_1 \leq \|\Pi_{x,y_1} - \nu_{f^{-1}(0)}\|_1 + \|\nu_{f^{-1}(0)} - \nu_{f^{-1}(1)}\|_1 + \|\Pi_{x,y_2} - \nu_{f^{-1}(1)}\|_1 \tag{5}$$

Combining (3), (4) and (5) we get

$$\|\nu_{f^{-1}(0)} - \nu_{f^{-1}(1)}\|_1 > 1.4.$$

Now to derive the contradiction, we define ν and ν_0 as following.

$$\nu := \Pr_{\substack{x\sim\mu \\ y\sim u}}[(x, y) \in Z] \cdot \nu_{f^{-1}(0)} + \Pr_{\substack{x\sim\mu \\ y\sim u}}[(x, y) \in f^{-1}(1)] \cdot \nu_{f^{-1}(1)}$$
$$+ \Pr_{\substack{x\sim\mu \\ y\sim u}}[(x, y) \notin f^{-1}(1),\ (x, y) \notin Z] \cdot \nu_{f^{-1}(*)}$$
$$\nu_0 := \nu_{f^{-1}(0)}$$

where $\nu_{f^{-1}(*)}$ denotes expected prior of transcripts from inputs that are neither in Z nor $f^{-1}(1)$. Then we have

$$\|\nu_0 - \nu\|_1 \geq \| \Pr_{\substack{x\sim\mu \\ y\sim u}}[(x, y) \in f^{-1}(1)] \cdot \left(\nu_{f^{-1}(0)} - \nu_{f^{-1}(1)}\right) \|_1$$
$$= \Pr_{\substack{x\sim\mu \\ y\sim u}}[(x, y) \in f^{-1}(1)] \cdot \|\nu_{f^{-1}(0)} - \nu_{f^{-1}(1)}\|_1 \geq \Omega\left(\Pr_{\substack{x\sim\mu \\ y\sim u}}[(x, y) \in f^{-1}(1)]\right)$$

since the difference is minimized when $\nu_{f^{-1}(*)} = \nu_{f^{-1}(0)}$. This via Pinsker's inequality implies

$$D(\nu_0||\nu) \geq \Omega\left(\Pr_{\substack{x\sim\mu \\ y\sim u}}[(x, y) \in f^{-1}(1)]^2\right).$$

while by our assumption, it must be the case $D(\nu_0||\nu) < 20b$ since $\Pr[(x, y) \in Z] > 1/2$.

To derive a contradiction, now we lower bound $\Pr_{\substack{x\sim\mu \\ y\sim u}}[(x, y) \in f^{-1}(1)]$. By Corollary 38,

$$\Pr_{\substack{x\sim\mu \\ y\sim u}}[(x, y) \in f^{-1}(1)]^2 = u(N(\mathsf{Supp}(\mu)))^2 \geq 2^{-2D(\mu||u)/\rho}$$

since $\mu = u|_{S_{\pi_a}}$. This implies that $\Pr_{\substack{x\sim\mu \\ y\sim u}}[(x, y) \in f^{-1}(1)]^2 \geq 2^{-2a/\rho}$ which is in contradiction to our setting of b. ◀

▶ **Claim 40** (Density). *Set $\rho = (\varepsilon \log d)^{1/c}$, and $d \geq (2 \cdot \log n)^{\frac{1}{1-\varepsilon}}$. Then*

$$\Pr_{(x,y)\sim u\times u}[f(x, y) = 0] \geq 1 - o(1/n)$$

Proof. We prove by bounding $\Pr_{(x,y)\sim u\times u}[\|x - y\|_\infty \leq c]$.

$$\Pr_{(x,y)\sim u\times u}[\|x - y\|_\infty < c] \leq \Pr_{x\sim u}[\|x\|_\infty < c] < \left(1 - 2^{-\rho^c}\right)^d = \left(1 - \frac{1}{d^\varepsilon}\right)^d \leq 2^{-d^{1-\varepsilon}} = o(1/n).$$

where the first inequality holds since $\max_{x\in\mathcal{X}} u(x) = u((0, \ldots, 0))$ by the property of u. ◀

Nearly Optimal Distinct Elements and Heavy Hitters on Sliding Windows

Vladimir Braverman[1]
Department of Computer Science, Johns Hopkins University, Baltimore, MD, USA
vova@cs.jhu.edu

Elena Grigorescu[2]
Department of Computer Science, Purdue University, West Lafayette, IN, USA
elena-g@purdue.edu

Harry Lang[3]
Department of Mathematics, Johns Hopkins University, Baltimore, MD, USA
hlang8@jhu.edu

David P. Woodruff[4]
School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA
dwoodruf@cs.cmu.edu

Samson Zhou[5]
Department of Computer Science, Purdue University, West Lafayette, IN, USA
samsonzhou@gmail.com

Abstract

We study the *distinct elements* and *ℓ_p-heavy hitters* problems in the *sliding window* model, where only the most recent n elements in the data stream form the underlying set. We first introduce the *composable histogram*, a simple twist on the exponential (Datar *et al.*, SODA 2002) and smooth histograms (Braverman and Ostrovsky, FOCS 2007) that may be of independent interest. We then show that the composable histogram along with a careful combination of existing techniques to track either the identity or frequency of a few specific items suffices to obtain algorithms for both distinct elements and ℓ_p-heavy hitters that are nearly optimal in both n and ϵ.

Applying our new composable histogram framework, we provide an algorithm that outputs a $(1+\epsilon)$-approximation to the number of distinct elements in the sliding window model and uses $\mathcal{O}\left(\frac{1}{\epsilon^2}\log n\log\frac{1}{\epsilon}\log\log n+\frac{1}{\epsilon}\log^2 n\right)$ bits of space. For ℓ_p-heavy hitters, we provide an algorithm using space $\mathcal{O}\left(\frac{1}{\epsilon^p}\log^2 n\left(\log^2\log n+\log\frac{1}{\epsilon}\right)\right)$ for $0<p\leq 2$, improving upon the best-known algorithm for ℓ_2-heavy hitters (Braverman *et al.*, COCOON 2014), which has space complexity $\mathcal{O}\left(\frac{1}{\epsilon^4}\log^3 n\right)$. We also show complementing nearly optimal lower bounds of $\Omega\left(\frac{1}{\epsilon}\log^2 n+\frac{1}{\epsilon^2}\log n\right)$ for distinct elements and $\Omega\left(\frac{1}{\epsilon^p}\log^2 n\right)$ for ℓ_p-heavy hitters, both tight up to $\mathcal{O}\left(\log\log n\right)$ and $\mathcal{O}\left(\log\frac{1}{\epsilon}\right)$ factors.

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

Keywords and phrases Streaming algorithms, sliding windows, heavy hitters, distinct elements

[1] This material is based upon work supported in part by the National Science Foundation under Grants No. 1447639, 1650041, and 1652257, Cisco faculty award, and by the ONR Award N00014-18-1-2364.
[2] Research supported in part by NSF CCF-1649515.
[3] This material is based upon work supported by the Franco-American Fulbright Commission. The author thanks INRIA (l'Institut national de recherche en informatique et en automatique) for hosting him during the writing of this paper.
[4] D. Woodruff would like to acknowledge the support by the National Science Foundation under Grant No. CCF-1815840.
[5] Research supported in part by NSF CCF-1649515.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018).
Editors: Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer; Article No. 7; pp. 7:1–7:22

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

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.7

Related Version A full version of the paper is available at [16], `https://arxiv.org/abs/1805.00212`.

1 Introduction

The streaming model has emerged as a popular computational model to describe large data sets that arrive sequentially. In the streaming model, each element of the input arrives one-by-one and algorithms can only access each element once. This implies that any element that is not explicitly stored by the algorithm is lost forever. While the streaming model is broadly useful, it does not fully capture the situation in domains where data is time-sensitive such as network monitoring [29, 30, 33] and event detection in social media [61]. In these domains, elements of the stream appearing more recently are considered more relevant than older elements. The *sliding window model* was developed to capture this situation [35]. In this model, the goal is to maintain computation on only the most recent n elements of the stream, rather than on the stream in its entirety. We call the most recent n elements *active* and the remaining elements *expired*. Any query is performed over the set of active items (referred to as the current window) while ignoring all expired elements.

The problem of identifying the number of distinct elements, is one of the foundational problems in the streaming model.

▶ **Problem 1** (Distinct elements). *Given an input S of elements in $[m]$, output the number of items i whose frequency f_i satisfies $f_i > 0$.*

The objective of identifying *heavy hitters*, also known as frequent items, is also one of the most well-studied and fundamental problems.

▶ **Problem 2** (ℓ_p-heavy hitters). *Given parameters $0 < \phi < \epsilon < 1$ and an input S of elements in $[m]$, output all items i whose frequency f_i satisfies $f_i \geq \epsilon(F_p)^{1/p}$ and no item i for which $f_i \leq (\epsilon - \phi)(F_p)^{1/p}$, where $F_p = \sum_{i\in[m]} f_i^p$. (The parameter ϕ is typically assumed to be at least $c\epsilon$ for some fixed constant $0 < c < 1$.)*

In this paper, we study the distinct elements and heavy hitters problems in the sliding window model. We show almost tight results for both problems, using several clean tweaks to existing algorithms. In particular, we introduce the composable histogram, a modification to the exponential histogram [35] and smooth histogram [19], that may be of independent interest. We detail our results and techniques in the following section, but defer complete proofs to the full version of the paper [16].

1.1 Our Contributions

Distinct elements.

An algorithm storing $\mathcal{O}\left(\frac{1}{\epsilon^2}\log n \log\frac{1}{\delta}(\log\frac{1}{\epsilon} + \log\log n)\right)$ bits in the insertion-only model was previously provided [53]. Plugging the algorithm into the smooth histogram framework of [19] yields a space complexity of $\mathcal{O}\left(\frac{1}{\epsilon^3}\log^3 n(\log\frac{1}{\epsilon} + \log\log n)\right)$ bits. We improve this significantly as detailed in the following theorem.

▶ **Theorem 1.** *Given $\epsilon > 0$, there exists an algorithm that, with probability at least $\frac{2}{3}$, provides a $(1+\epsilon)$-approximation to the number of distinct elements in the sliding window model, using $\mathcal{O}\left(\frac{1}{\epsilon^2}\log n \log\frac{1}{\epsilon}\log\log n + \frac{1}{\epsilon}\log^2 n\right)$ bits of space.*

Table 1 Our improvements for ℓ_2-heavy hitters and distinct elements in the sliding window model.

Problem	Previous Bound	New Bound
ℓ_2-heavy hitters	$\mathcal{O}\left(\frac{1}{\epsilon^4}\log^3 n\right)$ [15]	$\mathcal{O}\left(\frac{1}{\epsilon^2}\log^2 n\left(\log^2\log n+\log^2\frac{1}{\epsilon}\right)\right)$
Distinct elements	$\mathcal{O}\left(\frac{1}{\epsilon^3}\log^2 n+\frac{1}{\epsilon}\log^3 n\right)$ [53, 19]	$\mathcal{O}\left(\frac{1}{\epsilon^2}\log\frac{1}{\epsilon}\log n\log\log n+\frac{1}{\epsilon}\log^2 n\right)$

A known lower bound is $\Omega\left(\frac{1}{\epsilon^2}+\log n\right)$ bits [1, 50] for insertion-only streams, which is also applicable to sliding windows since the model is strictly more difficult. We give a lower bound for distinct elements in the sliding window model, showing that our algorithm is nearly optimal, up to $\log\frac{1}{\epsilon}$ and $\log\log n$ factors, in both n and ϵ.

▶ **Theorem 2.** *Let $0<\epsilon\leq\frac{1}{\sqrt{n}}$. Any one-pass streaming algorithm that returns a $(1+\epsilon)$-approximation to the number of distinct elements in the sliding window model with probability $\frac{2}{3}$ requires $\Omega\left(\frac{1}{\epsilon}\log^2 n+\frac{1}{\epsilon^2}\log n\right)$ bits of space.*

ℓ_p-heavy hitters.

We first recall in Lemma 16 a condition that allows the reduction from the problem of finding the ℓ_p-heavy hitters for $0<p\leq 2$ to the problem of finding the ℓ_2-heavy hitters. An algorithm of [12] allows us to maintain an estimate of F_2. However, observe in Problem 2 that an estimate for F_2 is only part of the problem. We must also identify which elements are heavy. First, we show how to use tools from [13] to find a superset of the heavy hitters. This alone is not enough since we may return false-positives (elements such that $f_i<(\epsilon-\phi)\sqrt{F_2}$). By keeping a careful count of the elements (shown in Section 4), we are able to remove these false-positives and obtain the following result, where we have set $\phi=\frac{11}{12}\epsilon$:

▶ **Theorem 3.** *Given $\epsilon>0$ and $0<p\leq 2$, there exists an algorithm in the sliding window model that, with probability at least $\frac{2}{3}$, outputs all indices $i\in[m]$ for which $f_i\geq\epsilon F_p^{1/p}$, and reports no indices $i\in[m]$ for which $f_i\leq\frac{\epsilon}{12}F_p^{1/p}$. The algorithm has space complexity (in bits) $\mathcal{O}\left(\frac{1}{\epsilon^p}\log^2 n\left(\log^2\log n+\log\frac{1}{\epsilon}\right)\right)$.*

Finally, we obtain a lower bound for ℓ_p-heavy hitters in the sliding window model, showing that our algorithm is nearly optimal (up to $\log\frac{1}{\epsilon}$ and $\log\log n$ factors) in both n and ϵ.

▶ **Theorem 4.** *Let $p>0$ and $\epsilon,\delta\in(0,1)$. Any one-pass streaming algorithm that returns the ℓ_p-heavy hitters in the sliding window model with probability $1-\delta$ requires $\Omega((1-\delta)\epsilon^{-p}\log^2 n)$ bits of space.*

More details are provided in Section 4 and Section 5.

By standard amplification techniques any result that succeeds with probability $\frac{2}{3}$ can be made to succeed with probability $1-\delta$ while multiplying the space and time complexities by $\mathcal{O}\left(\log\frac{1}{\delta}\right)$. Therefore Theorem 1 and Theorem 15 can be taken with regard to any positive probability of failure.

See Table 1 for a comparison between our results and previous work.

1.2 Our Techniques

We introduce a simple extension of the exponential and smooth histogram frameworks, which use several instances of an underlying streaming algorithm. In contrast with the existing frameworks where $\mathcal{O}(\log n)$ different sketches are maintained, we observe in Section 2 when the underlying algorithm has certain guarantees, then we can store these sketches more efficiently.

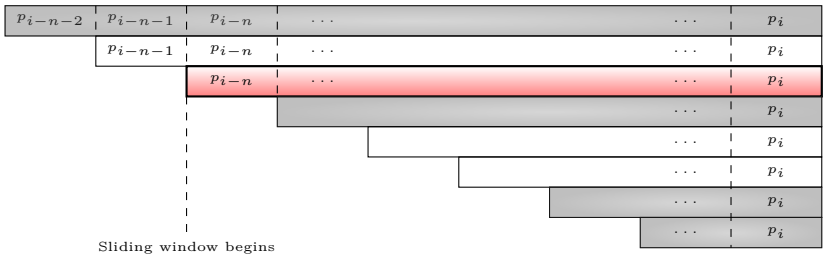


Figure 1 Each horizontal bar represents an instance of the insertion-only algorithm. The red instance represents the sliding window. Storing an instance beginning at each possible start point would ensure that the exact window is always available, but this requires linear space. To achieve polylogarithmic space, the histogram stores a strategically chosen set of $\mathcal{O}(\log n)$ instances (shaded grey) so that the value of f on any window can be $(1+\epsilon)$-approximated by its value on an adjacent window.

Algorithm 1 Input: a stream of elements $p_1, p_2, \ldots$ from $[m]$, a window length $n \geq 1$, error $\epsilon \in (0,1)$.

1: $T \leftarrow 0$
2: $i \leftarrow 1$
3: **loop**
4: Get p_i from stream
5: $T \leftarrow T+1$; $t_T \leftarrow i$; Compute $D(t_T)$, where $\hat{f}(D)$ is a $\left(1 \pm \frac{\epsilon}{4}\right)$-approximation of f.
6: **for** all $1 < j < T$ **do**
7: **if** $\hat{f}(D(t_{j-1} : t_T)) < \left(1 - \frac{\epsilon}{4}\right) \hat{f}(D(t_{j+1} : t_T))$ **then**
8: Delete t_j; update indices; $T \leftarrow T-1$
9: **if** $t_2 < i - n$ **then**
10: Delete t_1; update indices; $T \leftarrow T-1$
11: $i \leftarrow i+1$

Sketching Algorithms

Consider the sliding window model, where elements eventually expire. A very simple (but wasteful) algorithm is to simply begin a new instance of the insertion-only algorithm upon the arrival of each new element (Figure 1). The smooth histogram of [19], summarized in Algorithm 1, shows that storing only $\mathcal{O}(\log n)$ instances suffices.

Algorithm 1 may delete indices for either of two reasons. The first (Lines 9-10) is that the index simply expires from the sliding window. The second (Lines 7-8) is that the indices immediately before (t_{j-1}) and after (t_{j+1}) are so close that they can be used to approximate t_j.

For the distinct elements problem (Section 3), we first claim that a well-known streaming algorithm [6] provides a $(1+\epsilon)$-approximation to the number of distinct elements at all points in the stream. Although this algorithm is suboptimal for insertion-only streams, we show that it is amenable to the conditions of a composable histogram (Theorem 6). Namely, we show there is a sketch of this algorithm that is monotonic over suffixes of the stream, and thus there exists an efficient encoding that efficiently stores $D(t_i : t_{i+1})$ for each $1 \leq i < T$, which allows us to reduce the space overhead for the distinct elements problem.

For ℓ_2-heavy hitters (Section 4), we show that the ℓ_2 norm algorithm of [12] also satisfies the sketching requirement. Thus, plugging this into Algorithm 1 yields a method to maintain

an estimate of ℓ_2. Algorithm 2 uses this subroutine to return the identities of the heavy hitters. However, we would still require that all n instances succeed since even $\mathcal{O}(1)$ instances that fail adversarially could render the entire structure invalid by tricking the histogram into deleting the wrong information (see [19] for details). We show that the ℓ_2 norm algorithm of [12] actually contains additional structure that only requires the correctness of $\mathsf{polylog}(n)$ instances, thus improving our space usage.

1.3 Lower Bounds

Distinct elements.

To show a lower bound of $\Omega\left(\frac{1}{\epsilon}\log^2 n + \frac{1}{\epsilon^2}\log n\right)$ for the distinct elements problems, we show in Theorem 19 a lower bound of $\Omega\left(\frac{1}{\epsilon}\log^2 n\right)$ and we show in Theorem 22 a lower bound of $\Omega\left(\frac{1}{\epsilon^2}\log n\right)$. We first obtain a lower bound of $\Omega\left(\frac{1}{\epsilon}\log^2 n\right)$ by a reduction from the IndexGreater problem, where Alice is given a string $S = x_1 x_2 \cdots x_m$ and each x_i has n bits so that S has mn bits in total. Bob is given integers $i \in [m]$ and $j \in [2^n]$ and must determine whether $x_i > j$ or $x_i \leq j$.

Given an instance of the IndexGreater problem, Alice splits the data stream into blocks of size $\mathcal{O}\left(\frac{\epsilon n}{\log n}\right)$ and further splits each block into $\sqrt{n}$ pieces of length $(1+2\epsilon)^k$, padding the remainder of each block with zeros if necessary. For each $i \in [m]$, Alice encodes x_i by inserting the elements $\{0, 1, \ldots, (1+2\epsilon)^k - 1\}$ into piece x_i of block $(\ell - i + 1)$. Thus, the number of distinct elements in each block is much larger than the sum of the number of distinct elements in the subsequent blocks. Furthermore, the location of the distinct elements in block $(\ell - i + 1)$ encodes x_i, so that Bob can recover x_i and compare it with j.

We then obtain a lower bound of $\Omega\left(\frac{1}{\epsilon^2}\log n\right)$ by a reduction from the GapHamming problem. In this problem, Alice and Bob receive length-n bitstrings x and y, which have Hamming distance either at least $\frac{n}{2} + \sqrt{n}$ or at most $\frac{n}{2} - \sqrt{n}$, and must decide whether the Hamming distance between x and y is at least $\frac{n}{2}$. Recall that for $\epsilon \leq \frac{2}{\sqrt{n}}$, a $(1+\epsilon)$-approximation can differentiate between at least $\frac{n}{2} + \sqrt{n}$ and at most $\frac{n}{2} - \sqrt{n}$. We use this idea to show a lower bound of $\Omega\left(\frac{1}{\epsilon^2}\log n\right)$ by embedding $\Omega(\log n)$ instances of GapHamming into the stream. As in the previous case, the number of distinct elements corresponding to each instance is much larger than the sum of the number of distinct elements for the remaining instances, so that a $(1+\epsilon)$-approximation to the number of distinct elements in the sliding window solves the GapHamming problem for each instance.

Heavy hitters.

To show a lower bound on the problem of finding ℓ_p-heavy hitters in the sliding window model, we give a reduction from the AugmentedIndex problem. Recall that in the AugmentedIndex problem, Alice is given a length-n string $S \in \{1, 2 \ldots, k\}^n$ (which we write as $[k]^n$) while Bob is given an index $i \in [n]$, as well as $S[1, i-1]$, and must output the i^{th} symbol of the string, $S[i]$. To encode $S[i]$ for $S \in [k]^n$, Alice creates a data stream $a_1 \circ a_2 \circ \ldots \circ a_b$ with the invariant that the heavy hitters in the suffix $a_i \circ a_{i+1} \circ \ldots \circ a_b$ encode $S[i]$. Specifically, the heavy hitters in the suffix will be concentrated in the substream a_i and the identities of each heavy hitter in a_i gives a bit of information about the value of $S[i]$. To determine $S[i]$, Bob expires the elements $a_1, a_2, \ldots, a_{i-1}$ so all that remains in the sliding window is $a_i \circ a_{i+1} \circ \ldots \circ a_b$, whose heavy hitters encode $S[i]$.

1.4 Related Work

The study of the distinct elements problem in the streaming model was initiated by Flajolet and Martin [44] and developed by a long line of work [1, 45, 6, 38, 43]. Kane, Nelson, and Woodruff [53] give an optimal algorithm, using $\mathcal{O}\left(\frac{1}{\epsilon^2} + \log n\right)$ bits of space, for providing a $(1+\epsilon)$-approximation to the number of distinct elements in a data stream, with constant probability. Blasiok [9] shows that to boost this probability up to $1-\delta$ for a given $0 < \delta < 1$, the standard approach of running $\mathcal{O}\left(\log \frac{1}{\delta}\right)$ independent instances is actually sub-optimal and gives an optimal algorithm that uses $\mathcal{O}\left(\frac{\log \delta^{-1}}{\epsilon^2} + \log n\right)$ bits of space.

The ℓ_1-heavy hitters problem was first solved by Misra and Gries, who give a deterministic streaming algorithm using $\mathcal{O}\left(\frac{1}{\epsilon} \log n\right)$ space [59]. Other techniques include the CountMin sketch [32], sticky sampling [57], lossy counting [57], sample and hold [40], multi-stage bloom filters [21], sketch-guided sampling [54], and CountSketch [26]. Among the numerous applications of the ℓ_p-heavy hitters problem are network monitoring [37, 62], denial of service prevention [40, 4, 31], moment estimation [51], ℓ_p-sampling [60], finding duplicates [47], iceberg queries [41], and entropy estimation [22, 48].

A stronger notion of "heavy hitters" is the *ℓ_2-heavy hitters*. This is stronger than the ℓ_1-guarantee since if $f_i \geq \epsilon F_1$ then $f_i^2 \geq \epsilon^2 F_1^2 \geq \epsilon^2 F_2$ (and so $f_i \geq \epsilon\sqrt{F_2}$). Thus any algorithm that finds the ℓ_2-heavy hitters will also find all items satisfying the ℓ_1-guarantee. In contrast, consider a stream that has $f_i = \sqrt{m}$ for some i and $f_j = 1$ for all other elements j in the universe. Then the ℓ_2-heavy hitters algorithm will successfully identify i for some constant ϵ, whereas an algorithm that only provides the ℓ_1-guarantee requires $\epsilon = \frac{1}{\sqrt{n}}$, and therefore $\Omega(\sqrt{n} \log n)$ space for identifying i. Moreover, the ℓ_2-gaurantee is the best we can do in polylogarithmic space, since for $p > 2$ it has been shown that identifying ℓ_p-heavy hitters requires $\Omega(n^{1-2/p})$ bits of space [23, 5].

The most fundamental data stream setting is the insertion-only model where elements arrive one-by-one. In the insertion-deletion model, a previously inserted element can be deleted (each stream element is assigned $+1$ or -1, generalizing the insertion-only model where only $+1$ is used). Finally, in the sliding window model, a length n is given and the stream consists only of insertions; points expire after n insertions, meaning that (unlike the insertion-deletion model) the deletions are implicit. Letting $S = s_1, s_2, \ldots$ be the stream, at time t the frequency vector is built from the window $W = \{s_{t-(n-1)}, \ldots, s_t\}$ as the active elements, whereas items $\{s_1, \ldots, s_{t-n}\}$ are expired. The objective is to identify and report the "heavy hitters", namely, the items i for which f_i is large with respect to W.

Table 2 shows prior work for ℓ_2-heavy hitters in the various streaming models. A retuning of CountSketch in [63] solves the problem of ℓ_2-heavy hitters in $\mathcal{O}\left(\log^2 n\right)$ bits of space. More recently, [13] presents an ℓ_2-heavy hitters algorithm using $\mathcal{O}\left(\log n \log\log n\right)$ space. This algorithm is further improved to an $\mathcal{O}\left(\log n\right)$ space algorithm in [12], which is optimal.

In the insertion-deletion model, CountSketch is space optimal [26, 52], but the update time per arriving element is improved by [55]. Thus in some sense, the ℓ_2-heavy hitters problem is completely understood in all regimes except the sliding window model. We provide a nearly optimal algorithm for this setting, as shown in Table 2.

We now turn our attention to the sliding window model. The pioneering work by Datar *et al.* [35] introduced the exponential histogram as a framework for estimating statistics in the sliding window model. Among the applications of the exponential histogram are quantities such as count, sum of positive integers, average, and ℓ_p norms. Numerous other significant works include improvements to count and sum [46], frequent itemsets [28], frequency counts and quantiles [2, 56], rarity and similarity [36], variance and k-medians [3]

Table 2 Space complexity in bits of computing ℓ_2-heavy hitters in various streaming models. We write $n = |S|$ and to simplify bounds we assume $\log n = \mathcal{O}(\log m)$.

Model	Upper Bound	Lower Bound
Insertion-Only	$\mathcal{O}\left(\epsilon^{-2}\log n\right)$ [12]	$\Omega(\epsilon^{-2}\log n)$ [Folklore]
Insertion-Deletion	$\mathcal{O}\left(\epsilon^{-2}\log^2 n\right)$ [26]	$\Omega(\epsilon^{-2}\log^2 n)$ [52]
Sliding Windows	$\mathcal{O}\left(\epsilon^{-2}\log^2 n(\log\epsilon^{-1}+\log\log n)\right)$ [Theorem 15]	$\Omega(\epsilon^{-2}\log^2 n)$ [Theorem 4]

and other geometric problems [42, 25]. Braverman and Ostrovsky [19] introduced the smooth histogram as a framework that extends to smooth functions. [19] also provides sliding window algorithms for frequency moments, geometric mean and longest increasing subsequence. The ideas presented by [19] also led to a number of other results in the sliding window model [34, 17, 20, 18, 27, 39, 14]. In particular, Braverman *et al.* [15] provide an algorithm that finds the ℓ_2-heavy hitters in the sliding window model with $\phi = c\epsilon$ for some constant $c > 0$, using $\mathcal{O}\left(\frac{1}{\epsilon^4}\log^3 n\right)$ bits of space, improving on results by [49]. [7] also implements and provides empirical analysis of algorithms finding heavy hitters in the sliding window model. Significantly, these data structures consider insertion-only data streams for the sliding window model; once an element arrives in the data stream, it remains until it expires. It remains a challenge to provide a general framework for data streams that might contain elements "negative" in magnitude, or even strict turnstile models. For a survey on sliding window algorithms, we refer the reader to [11].

2 Composable Histogram Data Structure Framework

We first describe a data structure which improves upon smooth histograms for the estimation of functions with a certain class of algorithms. This data structure provides the intuition for the space bounds in Theorem 1. Before describing the data structure, we need the definition a smooth function.

▶ **Definition 5** ([19]). A function $f \geq 1$ is (α, β)-smooth if it has the following properties:

Monotonicity $f(A) \geq f(B)$ for $B \subseteq A$ (B is a suffix of A)

Polynomial boundedness There exists $c > 0$ such that $f(A) \leq n^c$.

Smoothness For any $\epsilon \in (0,1)$, there exists $\alpha \in (0,1)$, $\beta \in (0,\alpha]$ so that if $B \subseteq A$ and $(1-\beta)f(A) \leq f(B)$, then $(1-\alpha)f(A \cup C) \leq f(B \cup C)$ for any adjacent C.

We emphasize a crucial observation made in [19]. Namely, for $p > 1$, ℓ_p is a $\left(\epsilon, \frac{\epsilon^p}{p}\right)$-smooth function while for $p \leq 1$, ℓ_p is a (ϵ,ϵ)-smooth function.

Given a data stream $S = p_1, p_2, \ldots, p_n$ and a function f, let $f(t_1, t_2)$ represent f applied to the substream $p_{t_1}, p_{t_1+1}, \ldots, p_{t_2}$. Furthermore, let $D(t_1 : t_2)$ represent the data structure used to approximate $f(t_1, t_2)$.

▶ **Theorem 6.** *Let f be an (α,β)-smooth function so that $f = \mathcal{O}(n^c)$ for some constant c. Suppose that for all $\epsilon, \delta > 0$:*

(1) *There exists an algorithm $\mathcal{A}$ that maintains at each time t a data structure $D(1:t)$ which allows it to output a value $\hat{f}(1,t)$ so that*

$$\mathbf{Pr}\left[|\hat{f}(1,t) - f(1,t)| \leq \frac{\epsilon}{2} f(1,t), \text{ for all } 0 \leq t \leq n\right] \geq 1 - \delta.$$

(2) *There exists an algorithm $\mathcal{B}$ which, given $D(t_1 : t_i)$ and $D(t_i + 1 : t_{i+1})$, can compute $D(t_i : t_{i+1})$. Moreover, suppose storing $D(t_i : t_{i+1})$ uses $\mathcal{O}(g_i(\epsilon,\delta))$ bits of space.*

Then there exists an algorithm that provides a $(1+\epsilon)$*-approximation to* f *on the sliding window, using* $\mathcal{O}\left(\frac{1}{\beta}\log^2 n + \sum_{i=1}^{\frac{4}{\beta}\log n} g_i\left(\epsilon, \frac{\delta}{n}\right)\right)$ *bits of space.*

We remark that the first condition of Theorem 6 is called "strong tracking" and well-motivated by [10].

3 Distinct Elements

We first show that a well-known streaming algorithm that provides a $(1+\epsilon)$-approximation to the number of distinct elements actually also provides strong tracking. Although this algorithm uses $\mathcal{O}\left(\frac{1}{\epsilon^2}\log n\right)$ bits of space and is suboptimal for insertion-only streams, we show that it is amenable to the conditions of Theorem 6. Thus, we describe a few modifications to this algorithm to provide a $(1+\epsilon)$-approximation to the number of distinct elements in the sliding window model.

Define $\mathsf{lsb}(x)$ to be the 0-based index of least significant bit of a non-negative integer x in binary representation. For example, $\mathsf{lsb}(10) = 1$ and $\mathsf{lsb}(0) := \log(m)$ where we assume $\log(m) = \mathcal{O}(\log n)$. Let $S \subset [m]$ and $h : [m] \to \{0,1\}^{\log m}$ be a random hash function. Let $S_k := \{s \in S : \mathsf{lsb}(h(s)) \geq k\}$ so that $2^k|S_k|$ is an unbiased estimator for $|S|$. Moreover, for k such that $\mathbf{E}[S_k] = \Theta\left(\frac{1}{\epsilon^2}\right)$, the standard deviation of $2^k|S_k|$ is $\mathcal{O}(\epsilon|S|)$. Let $h_2 : [m] \to [B]$ be a pairwise independent random hash function with $B = \frac{100}{\epsilon^2}$. Let $\Phi_B(m)$ be the expected number of non-empty bins after m balls are thrown at random into B bins so that $\mathbf{E}[|h_2(S_k)|] = \Phi_B(|S_k|)$.

▶ **Fact 7.** $\Phi_m(t) = t\left(1 - \left(1 - \frac{1}{t}\right)^m\right)$

Blasiok provides an optimal algorithm for a constant factor approximation to the number of distinct elements with strong tracking.

▶ **Theorem 8** ([9])**.** *There is a streaming algorithm that, with probability* $1-\delta$*, reports a* $(1+\epsilon)$*-approximation to the number of distinct elements in the stream after every update and uses* $\mathcal{O}\left(\frac{\log\log n + \log\delta^{-1}}{\epsilon^2} + \log n\right)$ *bits of space.*

Thus we define an algorithm Oracle that provides a 2-approximation to the number of distinct elements in the stream after every update, using $\mathcal{O}(\log n)$ bits of space.

Since we can specifically track up to $\mathcal{O}\left(\frac{1}{\epsilon^2}\right)$ distinct elements, let us consider the case where the number of distinct elements is $\omega\left(\frac{1}{\epsilon^2}\right)$. Given access to Oracle to output an estimate K, which is a 2-approximation to the number of distinct elements, we can determine an integer $k > 0$ for which $\frac{K}{2^k} = \mathcal{O}\left(\frac{1}{\epsilon^2}\right)$. Then the quantity $2^k\Phi_B^{-1}(|h_2(S_k)|)$ provides both strong tracking as well as a $(1+\epsilon)$-approximation to the number of distinct elements:

▶ **Lemma 9** ([9])**.** *The median of* $\mathcal{O}(\log\log n)$ *estimators* $2^k\Phi_B^{-1}(|h_2(S_k)|)$ *is a* $(1+\epsilon)$*-approximation at all times for which the number of distinct elements is* $\Theta\left(\frac{2^k}{\epsilon^2}\right)$*, with constant probability.*

Hence, it suffices to maintain $h_2(S_i)$ for each $1 \leq i \leq \log m$, provided access to Oracle to find k, and $\mathcal{O}(\log\log n)$ parallel repetitions are sufficient to decrease the variance.

Indeed, a well-known algorithm for maintaining $h_2(S_i)$ simply keeps a $\log m \times \mathcal{O}\left(\frac{1}{\epsilon^2}\right)$ table T of bits. For $0 \leq i \leq \log n$, row i of the table corresponds to $h_2(S_i)$. Specifically, the bit in entry (i,j) of T corresponds to 0 if $h_2(s) \neq j$ for all $s \in S_i$ and corresponds to 1 if

there exists some $s \in S_i$ such that $h_2(s) = j$. Therefore, the table maintains $h_2(S_i)$, so then Lemma 9 implies that the table also gives a $(1+\epsilon)$-approximation to the number of distinct elements at all times, using $\mathcal{O}\left(\frac{1}{\epsilon^2}\log n\right)$ bits of space and access to Oracle. Then the total space is $\mathcal{O}\left(\frac{1}{\epsilon^2}\log n \log\log n\right)$ after again using $\mathcal{O}(\log\log n)$ parallel repetitions to decrease the variance.

Naïvely using this algorithm in the sliding window model would give a space usage dependency of $\mathcal{O}\left(\frac{1}{\epsilon^3}\log^2 n \log\log n\right)$. To improve upon this space usage, consider maintaining tables for substreams $(t_1, t), (t_2, t), (t_3, t), \ldots$ where $t_1 < t_2 < t_3 < \ldots < t$. Let T_i represent the table corresponding to substream (t_i, t). Since (t_{i+1}, t) is a suffix of (t_i, t), then the support of the table representing (t_{i+1}, t) is a subset of the support of the table representing (t_i, t). That is, if the entry (a, b) of T_{i+1} is one, then the entry (a, b) of T_i is one, and similarly for each $j < i$. Thus, instead of maintaining $\frac{1}{\epsilon}\log n$ tables of bits corresponding to each of the (t_i, t), it suffices to maintain a single table T where each entry represents the ID of the *last* table containing a bit of one in the entry. For example, if the entry (a, b) of T_9 is zero but the entry (a, b) of T_8 is one, then the entry (a, b) for T is 8. Hence, T is a table of size $\log m \times \mathcal{O}\left(\frac{1}{\epsilon^2}\right)$, with each entry having size $\mathcal{O}\left(\log\frac{1}{\epsilon} + \log\log n\right)$ bits, for a total space of $\mathcal{O}\left(\frac{1}{\epsilon^2}\log n\left(\log\frac{1}{\epsilon} + \log\log n\right)\right)$ bits. Finally, we need $\mathcal{O}\left(\frac{1}{\epsilon}\log^2 n\right)$ bits to maintain the starting index t_i for each of the $\frac{1}{\epsilon}\log n$ tables represented by T. Again using a number of repetitions, the space usage is $\mathcal{O}\left(\frac{1}{\epsilon^2}\log n\left(\log\frac{1}{\epsilon} + \log\log n\right)\log\log n + \frac{1}{\epsilon}\log^2 n\right)$.

Since this table is simply a clever encoding of the $\mathcal{O}\left(\frac{1}{\epsilon}\log n\right)$ tables used in the smooth histogram data structure, correctness immediately follows. We emphasize that the improvement in space follows from the idea of Theorem 6. That is, instead of storing a separate table for each instance of the algorithm in the smooth histogram, we instead simply keep the *difference* between each instance.

Finally, observe that each column in T is monotonically decreasing. This is because $S_k := \{s \in S : \mathsf{lsb}(h(s)) \geq k\}$ is a subset of S_{k-1}. Alternatively, if an item has been sampled to level k, it must have also been sampled to level $k-1$. Instead of using $\mathcal{O}\left(\log\frac{1}{\epsilon} + \log\log n\right)$ bits per entry, we can efficiently encode the entries for each column in T with the observation that each column is monotonically decreasing.

Proof of Theorem 1. Since the largest index of T_i is $i = \frac{1}{\epsilon}\log n$ and T has $\log m$ rows, the number of possible columns is $\binom{\frac{1}{\epsilon}\log n + \log m - 1}{\log m}$, which can be encoded using $\mathcal{O}\left(\log n \log\frac{1}{\epsilon}\right)$ bits. Correctness follows immediately from Lemma 9 and the fact that the estimator is monotonic. Again we use $\mathcal{O}\left(\frac{1}{\epsilon}\log^2 n\right)$ bits to maintain the starting index t_i for each of the $\frac{1}{\epsilon}\log n$ tables represented by T. As T has $\mathcal{O}\left(\frac{1}{\epsilon^2}\right)$ columns and accounting again for the $\mathcal{O}(\log\log n)$ repetitions to decrease the variance, the total space usage is $\mathcal{O}\left(\frac{1}{\epsilon^2}\log n \log\frac{1}{\epsilon}\log\log n + \frac{1}{\epsilon}\log^2 n\right)$ bits. ◀

4 ℓ_p Heavy Hitters

Subsequent analysis by Berinde *et al.* [8] proved that many of the classic ℓ_2-heavy hitter algorithms not only revealed the identity of the heavy hitters, but also provided estimates of their frequencies. Let $f_{tail(k)}$ be the vector f whose largest k entries are instead set to zero. Then an algorithm that, for each heavy hitter i, outputs a quantity $\hat{f}_i$ such that $|\hat{f}_i - f_i| \leq \epsilon||f_{tail(k)}||_1 \leq \epsilon||f||_1$ is said to satisfy the (ϵ, k)-*tail guarantee*. Jowhari *et al.* [52] show an algorithm that finds the ℓ_2-heavy hitters and satisfies the tail guarantee can also find the ℓ_p-heavy hitters. Thus, we first show results for ℓ_2-heavy hitters and then use this property to prove results for ℓ_p-heavy hitters.

To meet the space guarantees of Theorem 15, we describe an algorithm, Algorithm 2, that only uses the framework of Algorithm 1 to provide a 2-approximation of the ℓ_2 norm of the sliding window. We detail the other aspects of Algorithm 2 in the remainder of the section.

Recall that Algorithm 1 partitions the stream into a series of "jump-points" where f increases by a constant multiplicative factor. The oldest jump point is before the sliding window and initiates the active window, while the remaining jump points are within the sliding window. Therefore, it is possible for some items to be reported as heavy hitters after the first jump point, even though they do not appear in the sliding window at all! For example, if the active window has ℓ_2 norm 2λ, and the sliding window has ℓ_2 norm $(1+\epsilon)\lambda$, all $2\epsilon\lambda$ instances of a heavy hitter in the active window can appear before the sliding window even begins. Thus, we must prune the list containing all heavy hitters to avoid the elements with low frequency in the sliding window.

To account for this, we begin a counter for each element immediately after the element is reported as a potential heavy hitter. However, the counter must be sensitive to the sliding window, and so we attempt to use a smooth-histogram to count the frequency of each element reported as a potential heavy hitter. Even though the count function is (ϵ, ϵ) smooth, the necessity to track up to $\mathcal{O}\left(\frac{1}{\epsilon^2}\right)$ heavy hitters prevents us from being able to $(1+\epsilon)$-approximate the count of each element. Fortunately, a constant approximation of the frequency of each element suffices to reject the elements whose frequency is less than $\frac{\epsilon}{8}\ell_2$. This additional data structure improves the space dependency to $\mathcal{O}\left(\frac{1}{\epsilon^2}\right)$.

4.1 Background for Heavy Hitters

We now introduce concepts from [13, 12] to show the conditions of Theorem 6 apply, first describing an algorithm from [12] that provides a good approximation of F_2 at all times.

▶ **Theorem 10** (Remark 8 in [12]). *For any $\epsilon \in (0,1)$ and $\delta \in [0,1)$, there exists a one-pass streaming algorithm* Estimator *that outputs at each time t a value $\hat{F}_2^{(t)}$ so that*

$$\mathbf{Pr}\left[|\hat{F}_2^{(t)} - F_2^{(t)}| \le \epsilon F_2^{(t)}, \text{ for all } 0 \le t \le n\right] \ge 1-\delta,$$

and uses $\mathcal{O}\left(\frac{1}{\epsilon^2}\log m\left(\log\log m + \log\frac{1}{\epsilon}\right)\log\frac{1}{\delta}\right)$ bits of space and $\mathcal{O}\left(\left(\log\log m + \log\frac{1}{\epsilon}\right)\log\frac{1}{\delta}\right)$ update time.

The algorithm of Theorem 10 is a modified version of the AMS estimator [1] as follows. Given vectors Z_j of 6-wise independent Rademacher (i.e. uniform ± 1) random variables, let $X_j(t) = \left\langle Z_j, f^{(t)}\right\rangle$, where $f^{(t)}$ is the frequency vector at time t. Then [12] shows that $Y_t = \frac{1}{N}\sum_{j=1}^{N} X_{j,t}^2$ is a reasonably good estimator for F_2. By keeping $X_j(1,t_1), X_j(t_1+1,t_2), \ldots, X_j(t_i+1,t)$, we can compute $X_{j,t}$ from these sketches. Hence, the conditions of Theorem 6 are satisfied for Estimator, so Algorithm 1 can be applied to estimate the ℓ_2 norm. One caveat is that naïvely, we still require the probability of failure for each instance of Estimator to be at most $\frac{\delta}{\log n}$ for the data structure to succeed with probability at least $1-\delta$. We show in Appendix A that it suffices to only require the probability of failure for each instance of Estimator to be at most $\frac{\delta}{\text{polylog}\, n}$, thus incurring only $\mathcal{O}(\log\log n)$ additional space rather than $\mathcal{O}(\log n)$. We now refer to a heavy hitter algorithm from [12] that is space optimal up to $\log\frac{1}{\epsilon}$ factors.

▶ **Theorem 11** (Theorem 11 in [12]). *For any $\epsilon > 0$ and $\delta \in [0,1)$, there exists a one-pass streaming algorithm, denoted $(\epsilon,\delta) -$ BPTree, that with probability at least $(1-\delta)$, returns a set of $\frac{\epsilon}{2}$-heavy hitters containing every ϵ-heavy hitter and an approximate frequency for every item returned satisfying the $(\epsilon, 1/\epsilon^2)$-tail guarantee. The algorithm uses $\mathcal{O}\left(\frac{1}{\epsilon^2}\left(\log\frac{1}{\delta\epsilon}\right)(\log n + \log m)\right)$ bits of space and has $\mathcal{O}\left(\log\frac{1}{\delta\epsilon}\right)$ update time and $\mathcal{O}\left(\frac{1}{\epsilon^2}\log\frac{1}{\delta\epsilon}\right)$ retrieval time.*

Algorithm 2 ϵ-approximation to the ℓ_2-heavy hitters in a sliding window.

Input: A stream S of updates p_i for an underlying vector v and a window size n.
Output: A list including all elements i with $f_i \geq \epsilon\ell_2$ and no elements j with $f_j < \frac{\epsilon}{12}\ell_2$.

1: Maintain sketches $D(p_{t_1} : p_{t_2}), D(p_{t_2}+1 : p_{t_3}), \ldots, D(p_{t_{k-1}}+1 : p_{t_k})$ to estimate the ℓ_2 norm.
   ▷ Use Estimator and Algorithm 1 with parameters $\left(\frac{1}{2}, \frac{\delta}{2}\right)$ here.
2: Let A_i be the merged sketch $D(p_{t_i}+1 : p_{t_k})$.
3: For each merged sketch A_i, find a superset H_i of the $\frac{\epsilon}{16}$-heavy hitters.
   ▷ Use $\left(\frac{\epsilon}{16}, \frac{\delta}{2}\right) -$ BPTree here. (Theorem 11)
4: For each element in H_1, create a counter.
   ▷ Instantiate a $2 -$ SmoothCounter for each of the $\mathcal{O}\left(\frac{1}{\epsilon^2}\right)$ elements reported in H_1.
5: Let $\hat{\ell}_2$ be the estimated ℓ_2 norm of A_1.
   ▷ Output of Estimator on A_1. (Theorem 10)
6: For element $i \in H_1$, let $\hat{f}_i$ be the estimated frequency of i.
   ▷ Output by $2 -$ SmoothCounter. (Theorem 12)
7: Output any element i with $\hat{f}_i \geq \frac{1}{4}\epsilon\hat{\ell}_2$.

Observe that Theorem 10 combined with Theorem 6 already yields a prohibitively expensive $\frac{1}{\epsilon^3}$ dependency on ϵ. Thus, we can only afford to set ϵ to some constant in Theorem 10 and have a constant approximation to F_2 in the sliding window.

At the conclusion of the stream, the data structure of Theorem 6 has another dilemma: either it reports the heavy hitters for a set of elements $\mathcal{S}_1$ that is a superset of the sliding window or it reports the heavy hitters for a set of elements $\mathcal{S}_2$ that is a subset of the sliding window. In the former case, we can report a number of unacceptable false positives, elements that are heavy hitters for $\mathcal{S}_1$ but may not appear at all in the sliding window. In the latter case, we may entirely miss a number of heavy hitters, elements that are heavy hitters for the sliding window but arrive before $\mathcal{S}_2$ begins. Therefore, we require a separate smooth histogram to track the counter of specific elements.

▶ **Theorem 12.** *For any $\epsilon > 0$, there exists an algorithm, denoted $(1+\epsilon) -$ SmoothCounter, that outputs a $(1+\epsilon)$-approximation to the frequency of a given element in the sliding window model, using $\mathcal{O}\left(\frac{1}{\epsilon}(\log n + \log m)\log n\right)$ bits of space.*

The algorithm follows directly from Theorem 6 and the observation that ℓ_1 is (ϵ, ϵ)-smooth.

4.2 ℓ_2-Heavy Hitters Algorithm

We now prove Theorem 15 using Algorithm 2. We detail our ℓ_2-heavy hitters algorithm in full, using $\ell_2 = \sqrt{F_2}$ and ϵ-heavy hitters to refer to the ℓ_2-heavy hitters problem with parameter ϵ.

▶ **Lemma 13.** *Any element i with frequency $f_i > \epsilon\ell_2$ is output by Algorithm 2.*

▶ **Lemma 14.** *No element i with frequency $f_i < \frac{\epsilon}{12}\ell_2(W)$ is output by Algorithm 2.*

▶ **Theorem 15.** *Given $\epsilon, \delta > 0$, there exists an algorithm in the sliding window model (Algorithm 2) that with probability at least $1-\delta$ outputs all indices $i \in [m]$ for which $f_i \geq \epsilon\sqrt{F_2}$, and reports no indices $i \in [m]$ for which $f_i \leq \frac{\epsilon}{12}\sqrt{F_2}$. The algorithm has space complexity (in bits) $\mathcal{O}\left(\frac{1}{\epsilon^2}\log^2 n\left(\log^2\log n + \log\frac{1}{\epsilon}\right)\right)$.*

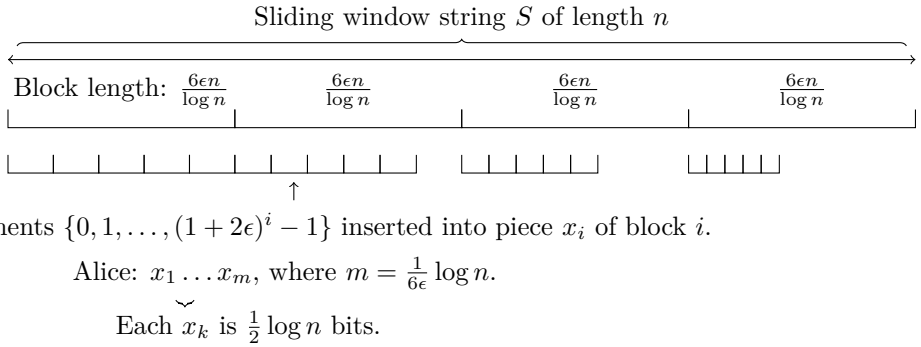


Figure 2 Construction of distinct elements instance by Alice. Pieces of block i have length $(1+2\epsilon)^i - 1$.

4.3 Extension to ℓ_p norms for $0 < p < 2$

To output a superset of the ℓ_p-heavy hitters rather than the ℓ_2-heavy hitters, recall that an algorithm provides the (ϵ, k)-tail guarantee if the frequency estimate $\hat{f}_i$ for each heavy hitter $i \in [m]$ satisfies $|\hat{f}_i - f_i| \leq \epsilon \cdot ||f_{tail(k)}||_1$, where $f_{tail(k)}$ is the frequency vector f in which the k most frequent entries have been replaced by zero. Jowhari *et al.* [52] show the impact of ℓ_2-heavy hitter algorithms that satisfy the tail guarantee.

▶ **Lemma 16** ([52]). *For any $p \in (0, 2]$, any algorithm that returns the $\epsilon^{p/2}$-heavy hitters for ℓ_2 satisfying the tail guarantee also finds the ϵ-heavy hitters for ℓ_p.*

The correctness of Theorem 3 immediately follows from Lemma 16 and Theorem 15.

5 Lower Bounds

5.1 Distinct Elements

To show a lower bound of $\Omega\left(\frac{1}{\epsilon}\log^2 n + \frac{1}{\epsilon^2}\log n\right)$ for the distinct elements problem, we show in Theorem 19 a lower bound of $\Omega\left(\frac{1}{\epsilon}\log^2 n\right)$ and we show in Theorem 22 a lower bound of $\Omega\left(\frac{1}{\epsilon^2}\log n\right)$. We first obtain a lower bound of $\Omega\left(\frac{1}{\epsilon}\log^2 n\right)$ by a reduction from the IndexGreater problem.

▶ **Definition 17.** In the IndexGreater problem, Alice is given a string $S = x_1 x_2 \cdots x_m$ of length mn, and thus each x_i has n bits. Bob is given integers $i \in [m]$ and $j \in [2^n]$. Alice is allowed to send a message to Bob, who must then determine whether $x_i > j$ or $x_i \leq j$.

Given an instance of the IndexGreater problem, Alice first splits the data stream into blocks of size $\mathcal{O}\left(\frac{\epsilon n}{\log n}\right)$. She further splits each block into $\sqrt{n}$ pieces of length $(1+2\epsilon)^k$, before padding the remainder of block $(\ell - k + 1)$ with zeros. To encode x_i for each $i \in [m]$, Alice inserts the elements $\{0, 1, \ldots, (1+2\epsilon)^k - 1\}$ into piece x_i of block $(\ell - i + 1)$, before padding the remainder of block $(\ell - k + 1)$ with zeros. In this manner, the number of distinct elements in each block dominates the number of distinct elements in the subsequent blocks. Moreover, the location of the distinct elements in block $(\ell - i + 1)$ encodes x_i, so that Bob can compare x_i to j. We formalize this argument in Appendix B.

▶ **Lemma 18.** *The one-way communication complexity of IndexGreater is $\Omega(nm)$ bits.*

▶ **Theorem 19.** *Let $p > 0$ and $\epsilon, \delta \in (0,1)$. Any one-pass streaming algorithm that returns a $(1+\epsilon)$-approximation to the number of distinct elements in the sliding window model with probability $\frac{2}{3}$ requires $\Omega\left(\frac{1}{\epsilon}\log^2 n\right)$ space.*

To obtain a lower bound of $\Omega\left(\frac{1}{\epsilon^2}\log n\right)$, we give a reduction from the GapHamming problem.

▶ **Definition 20** ([50])**.** In the GapHamming problem, Alice and Bob receive n bit strings x and y, which have Hamming distance either at least $\frac{n}{2}+\sqrt{n}$ or at most $\frac{n}{2}-\sqrt{n}$. Then Alice and Bob must decide which of these instances is true.

Chakrabarti and Regev show an optimal lower bound on the communication complexity of GapHamming.

▶ **Lemma 21** ([24])**.** *The communication complexity of GapHamming is $\Omega(n)$.*

Observe that a $(1+\epsilon)\frac{n}{2} \leq \frac{n}{2}+\sqrt{n}$ for $\epsilon \leq \frac{2}{\sqrt{n}}$ and thus a $(1+\epsilon)$-approximation can differentiate between at least $\frac{n}{2}+\sqrt{n}$ and at most $\frac{n}{2}-\sqrt{n}$. We use this idea to show a lower bound of $\Omega\left(\frac{1}{\epsilon^2}\log n\right)$ by embedding $\Omega(\log n)$ instances of GapHamming into the stream.

▶ **Theorem 22.** *Let $p > 0$ and $\epsilon, \delta \in (0,1)$. Any one-pass streaming algorithm that returns a $(1+\epsilon)$-approximation to the number of distinct elements in the sliding window model with probability $\frac{2}{3}$ requires $\Omega\left(\frac{1}{\epsilon^2}\log n\right)$ space for $\epsilon \leq \frac{1}{\sqrt{n}}$.*

Hence, Theorem 2 follows from Theorem 19 and Theorem 22.

5.2 ℓ_p-Heavy Hitters

To show a lower bound for the ℓ_p-heavy hitters problem in the sliding window model, we consider the following variant of the AugmentedIndex problem. Let k and n be positive integers and $\delta \in [0,1)$. Suppose the first player Alice is given a string $S \in [k]^n$, while the second player Bob is given an index $i \in [n]$, as well as $S[1, i-1]$. Alice sends a message to Bob, and Bob must output $S[i]$ with probability at least $1-\delta$.

▶ **Lemma 23** ([58])**.** *Even if Alice and Bob have access to a source of shared randomness, Alice must send a message of size $\Omega((1-\delta)n\log k)$ in a one-way communication protocol for the AugmentedIndex problem.*

We reduce the AugmentedIndex problem to finding the ℓ_p-heavy hitters in the sliding window model. To encode $S[i]$ for $S \in [k]^n$, Alice creates a data stream $a_1 \circ a_2 \circ \ldots \circ a_b$ with the invariant that the heavy hitters in the suffix $a_i \circ a_{i+1} \circ \ldots \circ a_b$ encodes $S[i]$. Thus to determine $S[i]$, Bob just needs to run the algorithm for finding heavy hitters on sliding windows and expire the elements $a_1, a_2, \ldots, a_{i-1}$ so all that remains in the sliding window is $a_i \circ a_{i+1} \circ \ldots \circ a_b$. We formally prove Theorem 4 in Appendix B.

References

1 Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. *J. Comput. Syst. Sci.*, 58(1):137–147, 1999. A preliminary version appeared in the Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing (STOC), 1996.

2 Arvind Arasu and Gurmeet Singh Manku. Approximate counts and quantiles over sliding windows. In *Proceedings of the Twenty-third ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 286–296, 2004.

3 Brian Babcock, Mayur Datar, Rajeev Motwani, and Liadan O'Callaghan. Maintaining variance and k-medians over data stream windows. In *Proceedings of the Twenty-Second ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS)*, pages 234–243, 2003.

4 Nagender Bandi, Divyakant Agrawal, and Amr El Abbadi. Fast algorithms for heavy distinct hitters using associative memories. In *27th IEEE International Conference on Distributed Computing Systems (ICDCS)*, page 6, 2007.

5 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. A preliminary version appeared in the Proceedings of the 43rd Symposium on Foundations of Computer Science (FOCS), 2002.

6 Ziv Bar-Yossef, T. S. Jayram, Ravi Kumar, D. Sivakumar, and Luca Trevisan. Counting distinct elements in a data stream. In *Randomization and Approximation Techniques, 6th International Workshop, RANDOM, Proceedings*, pages 1–10, 2002.

7 Ran Ben-Basat, Gil Einziger, Roy Friedman, and Yaron Kassner. Heavy hitters in streams and sliding windows. In *35th Annual IEEE International Conference on Computer Communications, INFOCOM*, pages 1–9, 2016.

8 Radu Berinde, Piotr Indyk, Graham Cormode, and Martin J. Strauss. Space-optimal heavy hitters with strong error bounds. *ACM Trans. Database Syst.*, 35(4):26:1–26:28, 2010. A preliminary version appeared in the Proceedings of the Twenty-Eigth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2009.

9 Jaroslaw Blasiok. Optimal streaming and tracking distinct elements with high probability. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 2432–2448, 2018.

10 Jaroslaw Blasiok, Jian Ding, and Jelani Nelson. Continuous monitoring of ℓ_{p} norms in data streams. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM*, pages 32:1–32:13, 2017.

11 Vladimir Braverman. Sliding window algorithms, 2016.

12 Vladimir Braverman, Stephen R. Chestnut, Nikita Ivkin, Jelani Nelson, Zhengyu Wang, and David P. Woodruff. Bptree: An ℓ_2 heavy hitters algorithm using constant memory. In *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS*, pages 361–376, 2017.

13 Vladimir Braverman, Stephen R. Chestnut, Nikita Ivkin, and David P. Woodruff. Beating countsketch for heavy hitters in insertion streams. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC*, pages 740–753, 2016.

14 Vladimir Braverman, Petros Drineas, Jalaj Upadhyay, and Samson Zhou. Numerical linear algebra in the sliding window model. *CoRR*, abs/1805.03765, 2018. `arXiv:1805.03765`.

15 Vladimir Braverman, Ran Gelles, and Rafail Ostrovsky. How to catch ℓ_2-heavy-hitters on sliding windows. *Theor. Comput. Sci.*, 554:82–94, 2014. A preliminary version appeared in the Proceedings of Computing and Combinatorics, 19th International Conference (COCOON), 2013.

16 Vladimir Braverman, Elena Grigorescu, Harry Lang, David P. Woodruff, and Samson Zhou. Nearly optimal distinct elements and heavy hitters on sliding windows. *CoRR*, abs/1805.00212, 2018. `arXiv:1805.00212`.

17 Vladimir Braverman, Harry Lang, Keith Levin, and Morteza Monemizadeh. Clustering on sliding windows in polylogarithmic space. In *35th IARCS Annual Conference on Foundation of Software Technology and Theoretical Computer Science, FSTTCS*, pages 350–364, 2015.

18 Vladimir Braverman, Harry Lang, Keith Levin, and Morteza Monemizadeh. Clustering problems on sliding windows. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 1374–1390, 2016.

19 Vladimir Braverman and Rafail Ostrovsky. Smooth histograms for sliding windows. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS) Proceedings*, pages 283–293, 2007.

20 Vladimir Braverman, Rafail Ostrovsky, and Alan Roytman. Zero-one laws for sliding windows and universal sketches. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM*, pages 573–590, 2015.

21 Yousra Chabchoub, Christine Fricker, and Hanene Mohamed. Analysis of a bloom filter algorithm via the supermarket model. In *21st International Teletraffic Congress, ITC*, pages 1–8, 2009.

22 Amit Chakrabarti, Graham Cormode, and Andrew McGregor. A near-optimal algorithm for estimating the entropy of a stream. *ACM Trans. Algorithms*, 6(3):51:1–51:21, 2010.

23 Amit Chakrabarti, Subhash Khot, and Xiaodong Sun. Near-optimal lower bounds on the multi-party communication complexity of set disjointness. In *18th Annual IEEE Conference on Computational Complexity*, pages 107–117, 2003.

24 Amit Chakrabarti and Oded Regev. An optimal lower bound on the communication complexity of gap-hamming-distance. *SIAM J. Comput.*, 41(5):1299–1317, 2012. A preliminary version appeared in the Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011.

25 Timothy M. Chan and Bashir S. Sadjad. Geometric optimization problems over sliding windows. *Int. J. Comput. Geometry Appl.*, 16(2-3):145–158, 2006. A preliminary version appeared in the Proceedings of Algorithms and Computation, 15th International Symposium (ISAAC), 2004.

26 Moses Charikar, Kevin C. Chen, and Martin Farach-Colton. Finding frequent items in data streams. *Theor. Comput. Sci.*, 312(1):3–15, 2004. A preliminary version appeared in the Proceedings of the Automata, Languages and Programming, 29th International Colloquium (ICALP), 2002.

27 Jiecao Chen, Huy L. Nguyen, and Qin Zhang. Submodular maximization over sliding windows. *CoRR*, abs/1611.00129, 2016.

28 Yun Chi, Haixun Wang, Philip S. Yu, and Richard R. Muntz. Catch the moment: maintaining closed frequent itemsets over a data stream sliding window. *Knowl. Inf. Syst.*, 10(3):265–294, 2006. A preliminary version appeared in the Proceedings of the 4th IEEE International Conference on Data Mining (ICDM), 2004.

29 Graham Cormode. The continuous distributed monitoring model. *SIGMOD Record*, 42(1):5–14, 2013.

30 Graham Cormode and Minos N. Garofalakis. Streaming in a connected world: querying and tracking distributed data streams. In *EDBT*, page 745, 2008.

31 Graham Cormode, Flip Korn, S. Muthukrishnan, and Divesh Srivastava. Finding hierarchical heavy hitters in streaming data. *TKDD*, 1(4):2:1–2:48, 2008.

32 Graham Cormode and S. Muthukrishnan. An improved data stream summary: the count-min sketch and its applications. *J. Algorithms*, 55(1):58–75, 2005. A preliminary version appeared in the Proceedings of the 6th Latin American Symposium (LATIN), 2004.

33 Graham Cormode and S. Muthukrishnan. What's new: finding significant differences in network data streams. *IEEE/ACM Transactions on Networking*, 13(6):1219–1232, 2005.

34 Michael S. Crouch, Andrew McGregor, and Daniel Stubbs. Dynamic graphs in the sliding-window model. In *Algorithms - ESA 2013 - 21st Annual European Symposium, Proceedings*, pages 337–348, 2013.

35 Mayur Datar, Aristides Gionis, Piotr Indyk, and Rajeev Motwani. Maintaining stream statistics over sliding windows. *SIAM J. Comput.*, 31(6):1794–1813, 2002. A preliminary version appeared in the Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), 2002.

36 Mayur Datar and S. Muthukrishnan. Estimating rarity and similarity over data stream windows. In *Algorithms - ESA 2002, 10th Annual European Symposium, Proceedings*, pages 323–334, 2002.

37 Erik D. Demaine, Alejandro López-Ortiz, and J. Ian Munro. Frequency estimation of internet packet streams with limited space. In *Algorithms - ESA, 10th Annual European Symposium, Proceedings*, pages 348–360, 2002.

38 Marianne Durand and Philippe Flajolet. Loglog counting of large cardinalities (extended abstract). In *Algorithms - ESA, 11th Annual European Symposium, Proceedings*, pages 605–617, 2003.

39 Alessandro Epasto, Silvio Lattanzi, Sergei Vassilvitskii, and Morteza Zadimoghaddam. Submodular optimization over sliding windows. In *Proceedings of the 26th International Conference on World Wide Web, WWW*, pages 421–430, 2017.

40 Cristian Estan and George Varghese. New directions in traffic measurement and accounting: Focusing on the elephants, ignoring the mice. *ACM Trans. Comput. Syst.*, 21(3):270–313, 2003.

41 Min Fang, Narayanan Shivakumar, Hector Garcia-Molina, Rajeev Motwani, and Jeffrey D. Ullman. Computing iceberg queries efficiently. In *VLDB'98, Proceedings of 24rd International Conference on Very Large Data Bases*, pages 299–310, 1998.

42 Joan Feigenbaum, Sampath Kannan, and Jian Zhang. Computing diameter in the streaming and sliding-window models. *Algorithmica*, 41(1):25–41, 2005.

43 Philippe Flajolet, Eric Fusy, Olivier Gandouet, and Frederic Meunier. Hyperloglog: the analysis of a near-optimal cardinality estimation algorithm. In *AofA: Analysis of Algorithms*, page 137–156, 2007.

44 Philippe Flajolet and G. Nigel Martin. Probabilistic counting. In *24th Annual Symposium on Foundations of Computer Science*, pages 76–82, 1983.

45 Phillip B. Gibbons and Srikanta Tirthapura. Estimating simple functions on the union of data streams. In *SPAA*, pages 281–291, 2001.

46 Phillip B. Gibbons and Srikanta Tirthapura. Distributed streams algorithms for sliding windows. In *SPAA*, pages 63–72, 2002.

47 Parikshit Gopalan and Jaikumar Radhakrishnan. Finding duplicates in a data stream. In *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 402–411, 2009.

48 Nicholas J. A. Harvey, Jelani Nelson, and Krzysztof Onak. Sketching and streaming entropy via approximation theory. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS*, pages 489–498, 2008.

49 Regant Y. S. Hung and Hing-Fung Ting. Finding heavy hitters over the sliding window of a weighted data stream. In *LATIN: Theoretical Informatics, 8th Latin American Symposium, Proceedings*, pages 699–710, 2008.

50 Piotr Indyk and David P. Woodruff. Tight lower bounds for the distinct elements problem. In *44th Symposium on Foundations of Computer Science (FOCS)*, pages 283–288, 2003.

51 Piotr Indyk and David P. Woodruff. Optimal approximations of the frequency moments of data streams. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC)*, pages 202–208, 2005.

52 Hossein Jowhari, Mert Saglam, and Gábor Tardos. Tight bounds for lp samplers, finding duplicates in streams, and related problems. In *Proceedings of the 30th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 49–58, 2011.

53 Daniel M. Kane, Jelani Nelson, and David P. Woodruff. An optimal algorithm for the distinct elements problem. In *Proceedings of the Twenty-Ninth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS*, pages 41–52, 2010.

54 Abhishek Kumar and Jun (Jim) Xu. Sketch guided sampling - using on-line estimates of flow size for adaptive data collection. In *INFOCOM 2006. 25th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies*, 2006.

55 Kasper Green Larsen, Jelani Nelson, Huy L. Nguyen, and Mikkel Thorup. Heavy hitters via cluster-preserving clustering. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS*, pages 61–70, 2016.

56 Lap-Kei Lee and H. F. Ting. A simpler and more efficient deterministic scheme for finding frequent items over sliding windows. In *Proceedings of the Twenty-Fifth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 290–297, 2006.

57 Gurmeet Singh Manku and Rajeev Motwani. Approximate frequency counts over data streams. *PVLDB*, 5(12):1699, 2012. A preliminary version appeared in the Proceedings of the 28th International Conference on Very Large Data Bases (VLDB), 2002.

58 Peter Bro Miltersen, Noam Nisan, Shmuel Safra, and Avi Wigderson. On data structures and asymmetric communication complexity. In *Proceedings of the Twenty-Seventh Annual ACM Symposium on Theory of Computing*, pages 103–111, 1995.

59 Jayadev Misra and David Gries. Finding repeated elements. *Sci. Comput. Program.*, 2(2):143–152, 1982.

60 Morteza Monemizadeh and David P. Woodruff. 1-pass relative-error ℓ_{p}-sampling with applications. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 1143–1160, 2010.

61 Miles Osborne, Sean Moran, Richard McCreadie, Alexander Von Lunen, Martin Sykora, Elizabeth Cano, Neil Ireson, Craig MacDonald, Iadh Ounis, Yulan He, Tom Jackson, Fabio Ciravegna, and Ann O'Brien. Real-time detection, tracking and monitoring of automatically discovered events in social media. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, 2014.

62 Subhabrata Sen and Jia Wang. Analyzing peer-to-peer traffic across large networks. *IEEE/ACM Trans. Netw.*, 12(2):219–232, 2004.

63 Mikkel Thorup and Yin Zhang. Tabulation-based 5-independent hashing with applications to linear probing and second moment estimation. *SIAM J. Comput.*, 41(2):293–331, 2012.

A Full Version

We show that the structure of the F_2 algorithm only requires the correctness of a specific $\mathcal{O}(\mathsf{polylog}\, n)$ algorithms in the data structure. Given a vector $v \in \mathbb{R}^m$, let $F_2(v) = v_1^2 + v_2^2 + \ldots + v_m^2$. Recall that the histogram creates a new algorithm each time a new element arrives in the data stream. Instead of requiring all n algorithms perform correctly, we show that it suffices to only require the correctness of a specific $\mathcal{O}(\mathsf{polylog}\, n)$ of these algorithms.

Let F be the value of F_2 on the most recent n elements. For the purpose of analysis, we say that an algorithm is *important* if it is still maintained within the histogram when its output is at least $\frac{F}{2\log n}$ and the algorithm never outputs anything greater than $8F\log^3 n$.

We first show that with high probability, all algorithms correctly maintain a $\log n$-approximation of the value of F_2 for the corresponding frequency vector. Conditioned on each algorithm correctly maintaining a $\log n$-approximation, we then show that $\mathcal{O}\left(\log^6 n\right)$ algorithms are important. Observe that an algorithm that reports a 2-approximation to F is important. Furthermore, we show that any algorithm that is not important cannot influence the output of the histogram, conditioned on each algorithm correctly maintaining a $\log n$-approximation. Thus, it suffices to require correctness of strong tracking on these $\mathcal{O}\left(\log^6 n\right)$ important algorithms and we apply a union bound over the $\mathcal{O}\left(\log^6 n\right)$ important algorithms to ensure correctness. Hence for each algorithm, we require the probability of failure to be at most $\mathcal{O}\left(\frac{\delta}{\log^6 n}\right)$ for the histogram to succeed with probability at least $1-\delta$.

▶ **Fact 24.** *Given m-dimensional vectors x, y, z with non-negative entries, then $F_2(x+y+z) - F_2(x+y) \geq F_2(x+z) - F_2(x)$.*

Although the number of algorithms in the histogram at any given moment is at most $\mathcal{O}(\log n)$, it may be possible that many algorithms have output at least $\frac{F}{2\log n}$ only to be deleted at some point in time. We now show that in a window of size $2n$, there are only $\mathcal{O}\left(\log^6 n\right)$ important algorithms.

▶ **Lemma 25.** *Conditioned on all algorithms in the stream correctly providing a $\log n$-approximation, then there are at most $\mathcal{O}\left(\log^6 n\right)$ important algorithms that begin in the most recent $2n$ elements.*

Proof. Let $s_1 < s_2 < \ldots < s_i$ be the starting points of important algorithms $A_1, A_2, \ldots, A_i$, respectively, that begin within the most recent $2n$ elements. For each $1 < j < i$, let t_j be the first time that algorithm A_j outputs a value that is at least $\frac{F}{2\log n}$. The idea is to show at the end of the stream, the elements between s_j and s_{j+1} are responsible for an increase in F_2 by at least $\frac{cF}{2\log^2 n}$ for all j. Since an algorithm is important if it never outputs anything greater than $8F\log^3 n$, then the F_2 value of the substream represented by the algorithm is at most $8F\log^4 n$, and it follows that $i = \mathcal{O}\left(\log^6 n\right)$.

Recall that to maintain the histogram, there exists a constant c such that whenever two adjacent algorithms have output within a factor of c, then we delete one of these algorithms. Hence, A_{j-1} must output a value that is at least $\frac{cF}{2\log n}$ at time t_j. Otherwise, the histogram would have deleted algorithm A_j before t_j, preventing A_j from being important. Conditioning on correctness of a $\log n$-approximation of all algorithms, the value of F_2 on the frequency vector from s_{j-1} to t_j is at least $\frac{cF}{2\log^2 n}$.

In other words, the elements from time s_{j-1} to s_j are responsible for a difference of at least $\frac{cF}{2\log^2 n}$ between the F_2 values of the substreams represented by A_{j-1} and A_j at time t_j. Thus by Fact 24, the difference between the F_2 values of the substreams represented by A_{j-1} and A_j at any time $t \geq t_j$ is at least $\frac{cF}{2\log^2 n}$. By induction, the value of F_2 on the substream from s_1 to t_j is at least $\frac{(j-1)cF}{2\log^2 n}$. Recall that the F_2 of the substream represented by any important algorithm is at most $8F\log^4 n$. Therefore, $i = \mathcal{O}\left(\log^6 n\right)$ and so at most $\mathcal{O}\left(\log^6 n\right)$ algorithms are important. ◀

▶ **Fact 26.** *For $x > 0$ and $a, b \geq 0$, $\frac{(x+a)^2}{x^2} \geq \frac{(x+a+b)^2}{(x+b)^2}$.*

▶ **Corollary 27.** *For $a_i, b_i, x_i \geq 0$ where $\sum x_i^2 > 0$, $\frac{\sum(x_i+a_i)^2}{\sum x_i^2} \geq \frac{\sum(x_i+a_i+b_i)^2}{(x_i+b_i)^2}$.*

▶ **Lemma 28.** *Conditioned on all algorithms in the stream correctly providing a $\log n$-approximation, then any algorithm that outputs a value that is at least $8F\log^3 n$ cannot delete an important algorithm that provides a 2-approximation to F.*

Proof. Note that any algorithm A that outputs a value that is at least $8F\log^3 n$ must represent a substream whose F_2 value is at least $8F\log^2 n$ at the end of the stream, assuming a $\log n$-approximation of all algorithms. Observe that the substream represented by an important algorithm B that provides a 2-approximation has F_2 value at most $2F$ at the end of the stream. By Corollary 27, the ratio between the F_2 values of the substreams represented by A and B must be at least $4\log^2 n$ at every previous point in time. Thus, if A and B always correctly maintain a $\log n$-approximation of the corresponding substreams, the ratio of the outputs between A and B is at least 4, so A will never cause the histogram data structure to delete B. ◀

Hence, it remains to show that with high probability, all algorithms correctly maintain a $\log n$-approximation of the value of F_2 for the corresponding frequency vector. Recall that Estimator from Theorem 10 uses an AMS sketch so that the resulting frequency of each element f_i is multiplied by a Rademacher random variable R_i.

▶ **Theorem 29** (Khintchine's inequality). *Let $R \in \{-1,1\}^m$ be chosen uniformly at random and $f \in \mathbb{R}^m$ be a given vector. Then for any even integer p, $\mathbf{E}\left[(\sum_{i=1}^m R_i f_i)^p\right] \leq \sqrt{p}^p ||f||_2^p$.*

Although we would like to apply Khintchine's inequality directly, the Rademacher random variables R_i used in Estimator are $\log n$-wise independent. Nevertheless, we can use independence to consider the $\log n$-th moment of the resulting expression.

▶ **Corollary 30.** *Let $z_1, z_2, \ldots, z_m \in \{-1,1\}$ be a set of $\log n$-wise independent random variables and $f \in \mathbb{R}^m$ be a given vector. Then for any even integer $p \leq \log n$, $\mathbf{E}\left[(\sum_{i=1}^m z_i f_i)^p\right] \leq \sqrt{p}^p ||f||_2^p$.*

We now show that each algorithm fails to maintain a $\log n$-approximation of the value of F_2 for the corresponding frequency vector only with negligible probability.

▶ **Lemma 31.** *Let $z_1, z_2, \ldots, z_m \in \{-1,1\}$ be a set of $\log n$-wise independent random variables and $f \in \mathbb{R}^m$ be a given vector. Then $\mathbf{Pr}\left[|\sum_{i=1}^m z_i f_i| \geq (\log n)||f||_2\right] \leq \frac{1}{\log n^{\sqrt{\log n}}}$.*

Proof. For the ease of notation, let $p = \log n$ be an even integer. Observe that

$$\mathbf{Pr}\left[\left|\sum_{i=1}^m z_i f_i\right| \geq (\log n)||f||_2\right] = \mathbf{Pr}\left[\left|\sum_{i=1}^m z_i f_i\right|^p \geq (\log n)^p ||f||_2^p\right].$$

By Markov's inequality, $\mathbf{Pr}\left[|\sum_{i=1}^m z_i f_i|^p \geq (\log n)^p ||f||_2^p\right] \leq \frac{\mathbf{E}\left[(\sum_{i=1}^m z_i f_i)^p\right]}{(\log n)^p ||f||_2^p}$. By Corollary 30, it follows that $\frac{\mathbf{E}\left[(\sum_{i=1}^m z_i f_i)^p\right]}{(\log n)^p ||f||_2^p} \leq \frac{\sqrt{p}^p ||f||_2^p}{(\log n)^p ||f||_2^p} = \frac{1}{\log n^{\sqrt{\log n}}}$. ◀

Therefore, with high probability, all algorithms correctly maintain a $\log n$-approximation of the value of F_2 for the corresponding frequency vector.

B Supplementary Proofs

Proof of Lemma 13. Since the ℓ_2 norm is a smooth function, and so there exists a smooth-histogram which is an $\left(\frac{1}{2}, \frac{\delta}{2}\right)$-estimation of the ℓ_2 norm of the sliding window by Theorem 6. Thus, $\frac{1}{2}\hat{\ell}_2(A_1) \leq \ell_2(W) \leq \frac{3}{2}\hat{\ell}_2(A_1)$. With probability $1 - \frac{\delta}{2}$, any element i whose frequency satisfies $f_i(W) \geq \epsilon \ell_2(W)$ must have $f_i(W) \geq \epsilon \ell_2(W) \geq \frac{1}{2}\epsilon\hat{\ell}_2(A_1)$ and is reported by $\left(\frac{\epsilon}{16}, \frac{\delta}{2}\right)$ − BPTree in Step 3.

Since BPTree is instantiated along with A_1, the sliding window may begin either before or after BPTree reports each heavy hitter. If the sliding window begins after the heavy hitter is reported, then all $f_i(W)$ instances are counted by SmoothCounter. Thus, the count of f_i estimated by SmoothCounter is at least $f_i(W) \geq \epsilon \ell_2(W) \geq \frac{1}{2}\epsilon\hat{\ell}_2(A_1)$, and so Step 7 will output i.

On the other hand, the sliding window may begin before the heavy hitter is reported. Recall that the BPTree algorithm identifies and reports an element when it becomes an $\frac{\epsilon}{16}$-heavy hitter with respect to the estimate of ℓ_2. Hence, there are at most $2 \cdot \frac{\epsilon}{16}\hat{\ell}_2(A_1) \leq \frac{1}{8}\epsilon\hat{\ell}_2(A_1)$ instances of an element appearing in the active window before it is reported by BPTree. Since $f_i(W) \geq \epsilon \ell_2(W) \geq \frac{1}{2}\epsilon\hat{\ell}_2(A_1)$, any element i whose frequency satisfies $f_i(W) \geq$

$\epsilon\ell_2(W)$ must have $f_i(W) \geq \frac{\epsilon}{2}\hat{\ell}_2(A_1)$ and therefore must have at least $\left(\frac{1}{2} - \frac{1}{8}\right)\epsilon\hat{\ell}_2(A_1) \geq \frac{1}{4}\epsilon\hat{\ell}_2(A_1)$ instances appearing in the stream after it is reported by BPTree. Thus, the count of f_i estimated by SmoothCounter is at least $\frac{1}{4}\epsilon\hat{\ell}_2(A_1)$, and so Step 7 will output i. ◀

Proof of Lemma 14. If i is output by Step 7, then $\hat{f}_i \geq \frac{1}{4}\epsilon\hat{\ell}_2(A_1)$. By the properties of SmoothCounter and Estimator, $f_i(W) \geq \frac{\hat{f}_i}{2} \geq \frac{1}{8}\epsilon\hat{\ell}_2(A_1) \geq \frac{1}{12}\ell_2(W)$, where the last inequality comes from the fact that $\ell_2(W) \leq \frac{3}{2}\hat{\ell}_2(A_1)$. ◀

Proof of Theorem 15. By Lemma 13 and Lemma 14, Algorithm 2 outputs all elements with frequency at least $\epsilon\ell_2(W)$ and no elements with frequency less than $\frac{\epsilon}{12}\ell_2(W)$. We now proceed to analyze the space complexity of the algorithm. Step 1 uses Algorithm 1 in conjunction with the Estimator routine to maintain a $\frac{1}{2}$-approximation to the ℓ_2-norm of the sliding window. By requiring the probability of failure to be $\mathcal{O}\left(\frac{\delta}{\text{polylog}n}\right)$ in Theorem 10 and observing that $\beta = \mathcal{O}(1)$ in Theorem 6 suffices for a $\frac{1}{2}$-approximation, it follows that Step 1 uses $\mathcal{O}\left(\log n(\log n + \log m \log^2 \log m)\right)$ bits of space. Since Step 3 runs an instance of BPTree for each of the at most $\mathcal{O}(\log n)$ buckets, then by Theorem 11, it uses $\mathcal{O}\left(\frac{1}{\epsilon^2}\left(\log\frac{1}{\delta\epsilon}\right)\log n(\log n + \log m)\right)$ bits of space.

Notice that BPTree returns a list of $\mathcal{O}\left(\frac{1}{\epsilon^2}\right)$ elements, by Theorem 11. By running SmoothCounter for each of these, Step 7 provides a 2-approximation to the frequency of each element after being returned by BPTree. By Theorem 12, Step 7 has space complexity (in bits) $\mathcal{O}\left(\frac{1}{\epsilon^2}(\log n + \log m)\log n\right)$. Assuming $\log m = \mathcal{O}(\log n)$, the algorithm uses $\mathcal{O}\left(\frac{1}{\epsilon^2}\log^2 n\left(\log^2 \log n + \log\frac{1}{\epsilon}\right)\right)$ bits of space. ◀

Proof of Theorem 3. By Theorem 11, BPTree satisfies the tail guarantee. Therefore by Lemma 16, it suffices to analyze the space complexity of finding the $\epsilon^{p/2}$-heavy hitters for ℓ_2. By Theorem 15, there exists an algorithm that uses $\mathcal{O}\left(\frac{1}{\epsilon^2}\log^2 n\left(\log^2 \log n + \log\frac{1}{\epsilon}\right)\right)$ bits of space to find the ϵ-heavy hitters for ℓ_2. Hence, there exists an algorithm that uses $\mathcal{O}\left(\frac{1}{\epsilon^p}\log^2 n\left(\log^2 \log n + \log\frac{1}{\epsilon}\right)\right)$ bits of space to find the ϵ-heavy hitters for ℓ_p, where $0 < p \leq 2$. ◀

Proof of Lemma 18. We show the communication complexity of IndexGreater through a reduction from the AugmentedIndex problem. Suppose Alice is given a string $S \in \{0,1\}^{nm}$ and Bob is given an index i along with the bits $S[1], S[2], \ldots, S[i-1]$. Then Bob's task in the AugmentedIndex problem is to determine $S[i]$.

Observe that Alice can form the string $T = x_1 x_2 \cdots x_m$ of length mn, where each x_k has n bits of S. Alice can then use the IndexGreater protocol and communicate to Bob a message that will solve the IndexGreater problem. Let $j = \left\lfloor \frac{i}{n} \right\rfloor$ so that the symbol $S[i]$ is a bit inside x_{j+1}. Then Bob constructs the string w by first concatenating the bits $S[jn+1], S[jn+2], \ldots, S[i-1]$, which he is given from the AugmentedIndex problem. Bob then appends a zero to w, and pads w with ones at the end, until w reaches n bits:

$$w = S[jn+1] \circ S[jn+2] \circ \cdots \circ S[i-1] \circ 0 \circ \underbrace{1 \circ 1 \circ \cdots \circ 1}_{\text{until } w \text{ has } n \text{ bits}} .$$

Bob takes the message from Alice and runs the IndexGreater protocol to determine whether $x_j > w$. Observe that by construction $x_j > w$ if and only if $S[i] = 1$. Thus, if the IndexGreater protocol succeeds, then Bob will have solved the AugmentedIndex problem, which requires communication complexity $\Omega(nm)$ bits. Hence, the communication complexity of IndexGreater follows. ◀

Proof of Theorem 19. We reduce a one-way communication protocol for IndexGreater to finding a $(1+\epsilon)$-approximation to the number of distinct elements in the sliding window model.

Let n be the length of the sliding window and suppose Alice receives a string $S = x_1 x_2 \dots x_\ell \in \{0,1\}^\ell$, where $\ell = \frac{1}{6\epsilon}\log n$ and each x_k has $\frac{1}{2}\log n$ bits. Bob receives an index $i \in [\ell]$ and an integer $j \in [\sqrt{n}]$. Suppose Alice partitions the sliding window into ℓ blocks, each of length $\frac{n}{\ell} = \frac{6\epsilon n}{\log n}$. For each $1 \le k \le \frac{1}{6\epsilon}\log n$, she further splits block $(\ell - k + 1)$ into $\sqrt{n}$ pieces of length $(1+2\epsilon)^k$, before padding the remainder of block $(\ell - k + 1)$ with zeros. Moreover, for piece x_k of block $(\ell - k + 1)$, Alice inserts the elements $\{0, 1, \dots, (1+2\epsilon)^k - 1\}$, before padding the remainder of block $(\ell - k + 1)$ with zeros. Hence, the sliding window contains all zeros, with the exception of the elements $\{0, 1, \dots, (1+2\epsilon)^k - 1\}$ appearing in piece x_k of block $(\ell - k + 1)$ for all $1 \le k \le \ell = \frac{1}{6\epsilon}\log n$. Note that $(1+2\epsilon)^k \le \sqrt[3]{n}$ and $x_k \le \sqrt{n}$ for all k, so all the elements fit within each block, which has length $\frac{6\epsilon n}{\log n}$. Finally, Alice runs the $(1+\epsilon)$-approximation distinct elements sliding window algorithm and passes the state to Bob. See Figure 2 for an example of Alice's construction.

Given integers $i \in [\ell]$ and $j \in [\sqrt{n}]$, Bob must determine if $x_i > j$. Thus, Bob is interested in x_i, so he takes the state of the sliding window algorithm, and inserts a number of zeros to expire each block before block i. Note that since Alice reversed the stream in her final step, Bob can do this by inserting $(\ell - i)\left(\frac{1}{2}\log n\right)$ number of zeros. Bob then inserts $(j-1)(1+2\epsilon)^i$ additional zeros, to arrive at piece j in block i. Since piece x_i contains $(1+2\epsilon)^i$ distinct elements and the remainder of the stream contains $(1+2\epsilon)^{i-1}$ distinct elements, then the output of the algorithm will decrease below $\frac{(1+2\epsilon)^i}{1+\epsilon}$ during piece x_i. Hence, if the output is less than $\frac{(1+2\epsilon)^i}{1+\epsilon}$ after Bob arrives at piece j, then $x_i \le j$. Otherwise, if the output is at least $\frac{(1+2\epsilon)^i}{1+\epsilon}$, then $x_i > j$. By the communication complexity of IndexGreater (Lemma 18), this requires space $\Omega\left(\frac{1}{\epsilon}\log^2 n\right)$. ◀

Proof of Theorem 22. We reduce a one-way communication protocol for the GapHamming problem to finding a $(1+\epsilon)$-approximation to the number of distinct elements in the sliding window model. For each $\frac{\log\frac{1}{\epsilon}}{2} \le i \le \frac{\log n - 1}{2}$, let $j = 2i$ and x_j and y_j each have length 2^j and (x_j, y_j) be drawn from a distribution such that with probability $\frac{1}{2}$, $\mathsf{HAM}(x_j, y_j) = (1+4\epsilon)2^{j-1}$ and otherwise (with probability $\frac{1}{2}$), $\mathsf{HAM}(x_j, y_j) = (1-4\epsilon)2^{j-1}$. Then Alice is given $\{x_j\}$ while Bob is given $\{y_j\}$ and needs to output $\mathsf{HAM}(x_j, y_j)$. For $\epsilon \le \frac{1}{\sqrt{n}}$, this is precisely the hard distribution in the communication complexity of GapHamming given by [24].

Let $a = \frac{\log\frac{1}{\epsilon}}{2}$ and $b = \frac{\log n - 1}{2}$. Let $w_{2k} = x_{2k}$ and let w_{2k-1} be a string of length 2^{2k-1}, all consisting of zeros. Suppose Alice forms the concatenated string $S = w_{2b} \circ w_{2b-1} \circ \dots \circ w_{2a+1} \circ w_{2a}$. Note that $\sum_{k=2a}^{2b} 2^k \le n$, so S has length less than n. Alice then forms a data stream by the following process. She initializes $k = 1$ and continuously increments k until $k = n$. At each step, if $S[k] = 0$ or k is longer than the length of S, Alice inserts a 0 into the data stream. Otherwise, if $S[k] = 1$, then Alice inserts k into the data stream. Meanwhile, Alice runs the $(1+\epsilon)$-approximation distinct elements sliding window algorithm and passes the state of the algorithm to Bob.

To find $\mathsf{HAM}(x_{2i}, y_{2i})$, Bob first expires $\left(\sum_{k=2i+1}^{2b} 2^k\right) - 2^{2i}$ elements by inserting zeros into the data stream. Similar to Alice, Bob initializes $k = 1$ and continuously increments k until $k = 2^{2i}$. At each step, if $y_{2i}[k] = 0$ (that is, the k^{th} bit of y_{2i} is zero), then Bob inserts a 0 into the data stream. Otherwise, if $y_{2i}[k] = 1$, then Bob inserts k into the data stream. At the end of this procedure, the sliding window contains all zeros, nonzero values corresponding to the nonzero indices of the string $x_{2i} \circ w_{2i-1} \circ x_{2i-2} \circ \dots \circ x_{2a+2} \circ w_{2a+1} \circ x_{2a}$, and nonzero values

corresponding to the nonzero indices of y_{2i}. Observe that each w_j solely consists of zeros and $\sum_{k=a}^{i-1} 2^{2k} < 2^{2i-1}$. Therefore, $\mathsf{HAM}(x_{2i}, y_{2i})$ is at least $(1-4\epsilon)2^{2i-1}$ while the number of distinct elements in the sliding window is at most $(1+4\epsilon)2^{2i}$ while the number of distinct elements in the suffix $x_{2i-2} \circ x_{2i-3} \cdots$ is at most $(1+\epsilon)2^{2i-2}$. Thus, a $(1+\epsilon)$-approximation to the number of distinct elements differentiates between $\mathsf{HAM}(x_{2i}, y_{2i}) = (1+4\epsilon)2^{2i-1}$ and $\mathsf{HAM}(x_{2i}, y_{2i}) = (1-4\epsilon)2^{2i-1}$.

Since the sliding window algorithm succeeds with probability $\frac{2}{3}$, then the GapHamming distance problem succeeds with probability $\frac{2}{3}$ across the $\Omega(\log n)$ values of i. Therefore, any $(1+\epsilon)$-approximation sliding window algorithm for the number of distinct elements that succeeds with probability $\frac{2}{3}$ requires $\Omega\left(\frac{1}{\epsilon^2}\log n\right)$ space for $\epsilon \leq \frac{1}{\sqrt{n}}$. ◀

Proof of Theorem 4. We reduce a one-way communication protocol for the AugmentedIndex problem to finding the ℓ_p heavy hitters in the sliding window model. Let $a = \frac{1}{2^p\epsilon^p}\log\sqrt{n}$ and $b = \log n$. Suppose Alice receives $S = [2^a]^b$ and Bob receives $i \in [b]$ and $S[1, i-1]$. Observe that each $S[i]$ is $\frac{1}{2^p\epsilon^p}\log\sqrt{n}$ bits and so $S[i]$ can be rewritten as $S[i] = w_1 \circ w_2 \circ \ldots \circ w_t$, where each $t = \frac{1}{2^p\epsilon^p}$ and so each w_i is $\log\sqrt{n}$ bits.

To recover $S[i]$, Alice and Bob run the following algorithm. First, Alice constructs data stream $A = a_1 \circ a_2 \circ \ldots \circ a_b$, which can be viewed as updates to an underlying frequency vector in $\mathbb{R}^n$. Each a_k consists of t updates, adding $2^{p(b-k)}$ to coordinates $v_1, v_2 \ldots, v_t$ of the frequency vector, where the binary representation of each $v_j \in [n]$ is the concatenation of the binary representation of j with the $\log\sqrt{n}$ bit string w_j. She then runs the sliding window heavy hitters algorithm and passes the state of the algorithm to Bob.

Bob expires all elements of the stream before a_i, runs the sliding window heavy hitters algorithm on the resulting vector, and then computes the heavy hitters. We claim that the algorithm will output t heavy hitters and by concatenating the last $\log\sqrt{n}$ bits of the binary representation of each of these heavy hitters, Bob will recover exactly $S[i]$. Observe that the ℓ_p norm of the underlying vector represented by $a_i \circ a_{i+1} \circ \ldots \circ a_b$ is exactly $\left(\frac{1}{2^p\epsilon^p}(1^p + 2^p + 4^p + \ldots + 2^{p(b-i)})\right)^{1/p} \leq \frac{1}{2\epsilon}2^{b-i+1} = \frac{1}{\epsilon}2^{b-i}$. Let $u_1, u_2 \ldots, u_t$ be the coordinates of the frequency vector incremented by Alice as part of a_i. Each coordinate u_j has frequency $2^{b-i} \geq \epsilon\left(\frac{1}{\epsilon}2^{b-i}\right)$, so that u_j is an ℓ_p-heavy hitter.

Moreover, the first $\log t$ bits of u_j encode $j \in [t]$ while the next $\log\sqrt{n}$ bits encode w_j. Thus, Bob identifies each heavy hitter and finds the corresponding $j \in [t]$ so that he can concatenate $S[i] = w_1 \circ w_2 \circ \ldots \circ w_t$. ◀

Survivable Network Design for Group Connectivity in Low-Treewidth Graphs

Parinya Chalermsook
Department of Computer Science, Aalto University, Espoo, Finland
parinya.chalermsook@aalto.fi

Syamantak Das
Indraprastha Institute of Information Technology Delhi, Delhi, India
syamantak@iiitd.ac.in
https://orcid.org/0000-0002-4393-8678

Guy Even
Tel-Aviv University, Tel-Aviv, Israel
guy@eng.tau.ac.il

Bundit Laekhanukit[1]
Max-Planck-Institut für Informatik, Saarücken, Germany and
Institute for Theoretical Computer Science, Shanghai University of Finance and Economics, Shanghai, China
blaekhan@mpi-inf.mpg.de
https://orcid.org/0000-0002-4476-8914

Daniel Vaz
Max-Planck-Institut für Informatik, Germany & Graduate School of Computer Science, Saarland University, Saarücken, Germany
ramosvaz@mpi-inf.mpg.de
https://orcid.org/0000-0003-2224-2185

Abstract

In the *Group Steiner Tree* problem (GST), we are given a (edge or vertex)-weighted graph $G = (V, E)$ on n vertices, together with a root vertex r and a collection of groups $\{S_i\}_{i\in[h]} : S_i \subseteq V(G)$. The goal is to find a minimum-cost subgraph H that connects the root to every group. We consider a fault-tolerant variant of GST, which we call Restricted (Rooted) Group SNDP. In this setting, each group S_i has a demand $k_i \in [k], k \in \mathbb{N}$, and we wish to find a minimum-cost subgraph $H \subseteq G$ such that, for each group S_i, there is a vertex in the group that is connected to the root via k_i (vertex or edge) disjoint paths.

While GST admits $O(\log^2 n \log h)$ approximation, its higher connectivity variants are known to be *Label-Cover* hard, and for the vertex-weighted version, the hardness holds even when $k = 2$ (it is widely believed that there is no subpolynomial approximation for the Label-Cover problem [Bellare et al., STOC 1993]). More precisely, the problem admits no $2^{\log^{1-\epsilon} n}$-approximation unless NP $\subseteq$ DTIME($n^{\text{polylog(n)}}$). Previously, positive results were known only for the edge-weighted version when $k = 2$ [Gupta et al., SODA 2010; Khandekar et al., Theor. Comput. Sci., 2012] and for a relaxed variant where k_i disjoint paths from r may end at different vertices in a group [Chalermsook et al., SODA 2015], for which the authors gave a bicriteria approximation. For $k \geq 3$, there is no non-trivial approximation algorithm known for edge-weighted Restricted Group SNDP, except for the special case of the relaxed variant on trees (folklore).

Our main result is an $O(\log n \log h)$ approximation algorithm for Restricted Group SNDP that runs in time $n^{f(k,w)}$, where w is the treewidth of the input graph. Our algorithm works for both edge and vertex weighted variants, and the approximation ratio nearly matches the lower

[1] ISF grant # 621/12, I-CORE grant # 4/11, NSF grant #CCF-1740425

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018).
Editors: Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer; Article No. 8; pp. 8:1–8:19

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

bound when k and w are constants. The key to achieving this result is a non-trivial extension of a framework introduced in [Chalermsook et al., SODA 2017]. This framework first embeds all feasible solutions to the problem into a dynamic program (DP) table. However, finding the optimal solution in the DP table remains intractable. We formulate a linear program relaxation for the DP and obtain an approximate solution via randomized rounding. This framework also allows us to systematically construct DP tables for high-connectivity problems. As a result, we present new exact algorithms for several variants of survivable network design problems in low-treewidth graphs.

2012 ACM Subject Classification Theory of computation → Routing and network design problems

Keywords and phrases Approximation Algorithms, Hardness of Approximation, Survivable Network Design, Group Steiner Tree

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.8

Related Version A full version appears at `https://arxiv.org/abs/1802.10403`.

Acknowledgements Parts of this work were done while Parinya Chalermsook, Bundit Laekhanukit and Daniel Vaz were visiting the Simons Institute for the Theory of Computing. It was partially supported by the DIMACS/Simons Collaboration on Bridging Continuous and Discrete Optimization through NSF grant #CCF-1740425. Parinya Chalermsook is supported by European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 759557) and the Academy of Finland Research Fellows Grant #310415. Parts of this work were done while Guy Even was visiting the Max-Planck-Institut für Informatik. Parts of this work were done while Syamantak Das was at Universität Bremen.

1 Introduction

Network design is an important subject in computer science and combinatorial optimization. The goal in network design is to build a network that meets some prescribed properties while minimizing the construction cost. *Survivable network design problems* (SNDP) are a class of problems where we wish to design a network that is resilient against link or node failures.

These problems have been phrased as optimization problems on graphs, where we are given an n-vertex (undirected or directed) graph $G = (V, E)$ with costs on edges or vertices together with a connectivity requirement $k : V \times V \to \mathbb{N}$. The goal is to find a minimum-cost subgraph $H \subseteq G$, such that every pair $u, v \in V$ of vertices are connected by $k(u, v)$ edge-disjoint (resp., openly vertex-disjoint) paths. In other words, we wish to design a network in which every pair of vertices remains connected (unless $k(u, v) = 0$), even after removing $k(u, v) - 1$ edges (or vertices). The *edge-connectivity* version of SNDP (EC-SNDP) models the existence of link failures and the *vertex-connectivity* (VC-SNDP) models the existence of both link and node failures. These two problems were known to be NP-hard and have received a lot of attention in the past decades (see, e.g., [26, 16, 31, 33]).

While VC-SNDP and EC-SNDP address the questions that arise from designing telecommunication networks, another direction of research focuses on the questions that arise from media broadcasting as in cable television or streaming services. In this case, we may wish to connect the global server to a single local server in each community, who will forward the

stream to all the clients in the area through their own local network. The goal here is slightly different from the usual SNDP, as it is not required to construct a network that spans every client; instead, we simply need to choose a local server (or representative), which will take care of connecting to other clients in the same group. This scenario motivates the *Group Steiner Tree* problem (GST) and its fault-tolerant variant, the *Rooted Group SNDP*.

In Rooted Group SNDP, we are given a graph $G = (V, E)$ with costs on edges or vertices, a root vertex r, and a collection of subsets of vertices called *groups*, $S_1, \ldots, S_h$, together with connectivity demands $k_1, \ldots, k_h \in [k]$, $k \in \mathbb{N}$. The goal in this problem is to find a minimum cost subgraph $H \subseteq G$ such that H has k_i edge-disjoint (or openly vertex-disjoint) paths connecting the root vertex r to some vertex $v_i \in S_i$, for all $i \in [h]$. In other words, we wish to choose one representative from each group and find a subgraph of G such that each representative is k-edge-(or vertex)-connected to the root.

When $k = 1$, the problem becomes the well-known *Group Steiner Tree* (GST) problem. Here, we are given a graph $G = (V, E)$ with edge or vertex costs, a root r and a collection of subsets of vertices called groups, $S_1, \ldots, S_h \subseteq V$, and the goal is to find a minimum-cost subgraph $H \subseteq G$ that has a path to some vertex in each S_i, for $i \in [h]$. The GST problem is known to admit an $O(\log^3 n)$-approximation algorithm [22] and cannot be approximated to a factor of $\log^{2-\epsilon} n$ unless NP $\subseteq$ ZPTIME$(n^{\text{polylog(n)}})$ [25].

The Rooted Group SNDP generalizes GST to handle fault tolerance. The case where $k = 2$ is studied in [27, 23], culminating in the $\tilde{O}(\log^4 n)$-approximation algorithm for the problem. For $k \geq 3$, there is no known non-trivial approximation algorithm. It is known among the experts that this problem is at least as hard as the *Label-Cover* problem[2].

Chalermsook, Grandoni and Laekhanukit [13] studied a relaxed version of the problem in which we are not restricted to connect to a single vertex in each group and thus need only k_i edge-disjoint paths connecting the root vertex to the whole group S_i. Despite being a relaxed condition, the problem remains as hard as the Label-Cover problem, and they only managed to design a bicriteria approximation algorithm.

To date, there is no known bicriteria or even sub-exponential-time poly-logarithmic approximation for Rooted Group SNDP when $k \geq 3$. The following is an intriguing open question:

> What are the settings (i.e., ranges of k or graph classes) in which Rooted Group SNDP admits a poly-logarithmic approximation?

In this paper, we focus on developing algorithmic techniques to approach the above question. We design poly-logarithmic algorithms for a special class of graphs – graphs with bounded treewidth – in the hope that it will shed some light towards solving the problem on a more general class of graphs, for instance, planar graphs (this is the case for the Steiner tree problem, where a sub-exponential-time algorithm for planar graphs is derived via decomposition into low-tree width instances [32]).

Our main technical building block is a dynamic program (DP) that solves rooted versions of EC-SNDP and VC-SNDP in bounded-treewidth graphs. However, a straightforward DP computation is not applicable for Restricted Rooted Group SNDP, simply because the problem is NP-hard on trees (so it is unlikely to admit a polynomial-size DP-table). Hence, we "embed" the DP table into a tree and apply a polylogarithmic approximation algorithm using randomized rounding of a suitable LP-formulation by Chalermsook et al. [12]. We remark that when the cost is polynomially bounded (e.g., in the Word RAM model with words of size

[2] The hardness for the case of directed graph was shown in [27], but it is not hard to show the same result for undirected graphs.

$O(\log n)$), polynomial-time algorithms for EC-SNDP and VC-SNDP follows from *Courcelle's Theorem* [17, 6] (albeit, with much larger running time). However, employing the theorem as a black-box does not allow us to design approximation algorithms for Restricted Rooted Group SNDP.

To avoid confusion between the relaxed and restricted version of Rooted Group SNDP (usually having the same name in literature), we refer to our problem as Restricted Group SNDP. (For convenience, we also omit the word "rooted".)

1.1 Related Work

SNDP problems on restricted graph classes have also been studied extensively. When $k = 1$, the problems are relatively well understood. Approximation algorithms and PTAS have been developed for many graph classes: low-treewidth graphs [1, 18], metric-cost graphs [14], Euclidean graphs [9], planar graphs [8], and graphs of bounded genus [7]. However, when $k \geq 2$, the complexity of these problems remains wide open. Borradaile et al. [7, 10] showed an algorithm for $k = 1, 2, 3$ on planar graphs, but under the assumption that one can buy multiple copies of edges (which they called *relaxed connectivity* setting). Without allowing multiplicity, very little is known when $k \geq 2$: Czumaj et al. [19] showed a PTAS for $k = 2$ in unweighted planar graphs, and Berger et al. [3] showed an exact algorithm running in time $2^{O(w^2)}n$ for the uniform demand case (i.e., $k(u, v) = 2$ for all pairs (u, v)). Thus, without the relaxed assumption, with non-uniform demands or $k > 2$, the complexity of SNDP problems on bounded-treewidth graphs and planar graphs is not adequately understood.

The technique of formulating an LP from a DP table has been used in literature. It is known that any (discrete) DP can be formulated as an LP, which is integral [30]. (For Stochastic DP, please see, e.g., [29, 21, 11, 20].) However, the technique of producing a tree structure out of a DP table is quite rare. Prior to this paper the technique of rounding LP via a tree structure was used in [24] to approximate the Sparsest-Cut problem. The latter algorithm is very similar to us. However, while we embed a graph into a tree via a DP table, their algorithm works directly on the tree decomposition. We remark that our technique is based on the previous work in [12] with almost the same set of authors.

1.2 Hardness of Approximating Restricted Group SNDP

As mentioned, it is known among the experts that vertex-cost variant of the Restricted Group SNDP has a simple reduction for the *Label-Cover* problem and more generally, the k-*Constraint Satisfaction* problem (k-CSP). The original construction was given by Khandekar, Kortsarz and Nutov [27] for the Restricted Group SNDP on directed graphs. However, the same construction applies for the Vertex-Weighted Restricted Group SNDP. The k-CSP hardness implies that even for $k_i \in \{0, 2\}$, Vertex-Weighted Restricted Group SNDP cannot be approximated to within a factor of $2^{\log^{1-\epsilon} n}$, for any $\epsilon > 0$, unless NP $\subseteq$ DTIME($n^{\text{polylog(n)}}$), and the approximation hardness is conjectured to be polynomial on n, say n^{δ} for some $0 < \delta < 1$, under the *Sliding Scale Conjecture* [2]. So far, we do not know of any non-trivial approximation for this problem for $k \geq 2$.

To be formal, the approximation hardness of Vertex-Weighted Restricted Group SNDP is stated below. While the hardness result is considered a folklore, we are aware that this fact might not be clear for the readers. We provide the proof sketch in the full version of the paper.

▶ **Theorem 1** (Folklore). *Consider the Vertex-Weighted Restricted Group SNDP problem when k_i are bounded for all i. The problem admits no $2^{\log^{1-\epsilon} n}$-approximation algorithm, for any constant $\epsilon > 0$, unless* NP $\subseteq$ DTIME($n^{\text{polylog(n)}}$). *Moreover, assuming the Sliding*

Scale Conjecture, there exists a constant $0 < \delta < 1$ *such that the problem admits no* n^{δ}*-approximation algorithm.*

The edge-cost variant has been studied in [27, 23], and a polylogarithmic approximation is known for the case $k = 2$ [27]. For $k > 3$, there is no known non-trivial approximation algorithm. The relaxed variant where the k disjoint paths from the root may end at different vertices in each group S_i has also been studied in [27, 23]. Chalermsook, Grandoni and Laekhanukit proposed a bicriteria approximation algorithm for the Relaxed Restricted Group SNDP [13]; however, their technique is not applicable for the restricted version. Note that the hardness of the edge-cost variant of Relaxed Restricted Group SNDP is $k^{1/6-\epsilon}$, for any $\epsilon > 0$ [13]. It is not hard to construct the same hardness result for Restricted Group SNDP. We believe that Restricted Group SNDP is strictly harder than the relaxed variant.

1.3 Our Results & Techniques

Our main result is the following approximation result for Restricted Group SNDP.

▶ **Theorem 2.** *There is a randomized algorithm that runs in time* $n^{f(w,k)}$*, for some function* f*, and returns, with high probability, a feasible solution to Restricted Group SNDP that has expected cost at the most* $O(\log n \log h)$ *times the cost of an optimal solution. Moreover, the algorithm works for both edge and vertex weighted variants.*

The proof of this theorem relies on the technique introduced in [12]. We give an overview of this technique and highlight how this paper departs from it.

In short, this technique "bridges" the ideas of dynamic program (DP) and randomized LP rounding in two steps[3]. Let us say that we would like to approximate optimization problem. In the first step, a "nice" DP table that captures the computation of the optimal solution is created, and there is a 1-to-1 correspondence between the DP solution and the solution to the problem. However, since the problem is NP-hard (in our case, even hard to approximate to within some poly-logarithmic factor), we could not follow the standard bottom-up computation of DP solutions. The idea of the second step is to instead write an LP relaxation that captures the computation of the optimal DP solution, and then use a randomized dependent rounding to get an approximate solution instead; the randomized rounding scheme is simply the well-known GKR rounding [22]. Roughly speaking, the size of the DP table is $n \cdot w^{O(w)}$, while the LP relaxation has $n^{O(w \log w)}$ variables and constraints, so we could get an $O(\log n \log h)$ approximation in time $n^{O(w \log w)}$.

The main technical hurdle that prevents us from using this technique to Restricted Group SNDP directly is that there was no systematic way to generate a "good" DP table for arbitrary connectivity demand k. (The previous result was already complicated even for $k = 1$.) This is where we need to depart from the previous work. We devise a new concept that allows us to systematically create such a DP table for any connectivity demand k. Our DP table has size $n \cdot f(k, w)$ for some function k and w, and it admits the same randomized rounding scheme in time $n^{g(k,w)}$, therefore yielding the main result.

As by-products, we obtain new DP algorithms for some well-studied variants of SNDP, whose running time depends on the treewidth of the input graph (in particular, $n \cdot f(k, w)$).

[3] One may view our result as a "tree-embedding" type result. Please see [12] for more discussion along this line. Here we choose to present our result in the viewpoint of DP & LP.

Subset connectivity problems: Subset k-Connectivity is a well-studied SNDP problem (Subset k-EC and Subset k-VC for edge and vertex connectivity, respectively). In this setting, all pairs of terminals have the same demands, i.e., $k(u,v) = k$ for all $u, v \in T$. This is a natural generalization of Steiner tree that has received attention [14, 31, 28].

▶ **Theorem 3.** *There are exact algorithms for Subset k-EC and Subset k-VC that run in time $f_1(k,w)n$ for some function f_1. This result holds for vertex- or edge-costs.*

Rooted SNDP: Another setting that has been studied in the context of vertex connectivity requirements is the Rooted SNDP [15, 31]. In this problem, there is a designated terminal vertex $r \in T$, and all positive connectivity requirements are enforced only between r and other terminals, i.e. $k(u,v) > 0$ only if $u = r$ or $v = r$. For the edge connectivity setting, Rooted SNDP captures Subset k-EC[4].

▶ **Theorem 4.** *There are exact algorithms for* Rooted EC-SNDP *and* Rooted VC-SNDP *that run in time $f_2(k,w)n$ for some function f_2. This result holds for costs on vertices or edges.*

Further technical overview: Let us illustrate how our approach is used to generate the DP table, amenable for randomized rounding. The following discussion assumes a certain familiarity with the notion of treewidth and DP algorithms in low-treewidth graphs.

Given graph $G = (V, E)$, let $\mathcal{T}$ be a tree decomposition of G having width w, i.e. each node $t \in V(\mathcal{T})$ corresponds to a *bag* $X_t \subseteq V(G) : |X_t| \leq w$. Let $\mathcal{T}_t$ denote the subtree of $\mathcal{T}$ rooted at t. For each $t \in V(\mathcal{T})$, let G_t denote the subgraph induced on all bags belonging to the subtree of $\mathcal{T}$ rooted at t, i.e. $G_t = G[\bigcup_{t \in \mathcal{T}_t} X_t]$. At a high level, DPs for minimization problems in low-treewidth graphs proceed as follows. Let Π denote the set of all possible profiles defined on the basis of some relevant property of the graph. For each node $t \in V(\mathcal{T})$, there is a profile $\pi_t \in \Pi$, and we define a DP cell $c[t, \pi_t]$ for each possible such profile, which stores the minimum-cost of a solution (a subgraph of G_t) that is consistent with the profile π_t. Then, a recursive rule is applied: Let t', t'' be the left and right children of t in $\mathcal{T}$ respectively. The DP makes a choice to "buy" a subset of edges $Y \subseteq E(G[X_t])$ (that appear in bag X_t) and derives the cost by minimizing over all profiles $\pi_{t'}, \pi_{t''}$ that are "consistent" with π_t:

$$c[t, \pi_t] = \min_{\pi_{t'}, \pi_{t''}, Y : (\pi_{t'}, \pi_{t''}, Y) \bowtie \pi_t} (\text{cost}(Y) + c[t', \pi_{t'}] + c[t'', \pi_{t''}])$$

where the sign $(\pi_{t'}, \pi_{t''}, Y) \bowtie \pi_t$ represents the notion of consistency between the profiles. Different optimization problems have different profiles and consistency rules. Often, consistency rules that are designed for connectivity-1 problems (such as Steiner tree) are not easily generalizable to higher connectivity problems (such as SNDP).

In this paper, we devise a new consistency rule (abbreviated by $\fallingdotseq$) for checking "reachability" (or connectivity 1) in a graph, which allows for easy generalization to handle high connectivity problems.

Roughly speaking, our consistency rule $\fallingdotseq$ solves the Steiner tree problem (connectivity-1 problem). To solve a connectivity-k problem, we have a DP cell $c[t, \vec{\pi}]$ for each $\vec{\pi} \in \Pi^k$. Then the consistency check is a "direct product" test for all coordinates, i.e.,

$$(\vec{\pi}_{t'}, \vec{\pi}_{t''}, \vec{Y}) \fallingdotseq^k \vec{\pi}_t \iff (\forall j \in [k])(\pi_{t',j}, \pi_{t'',j}, Y_j) \fallingdotseq \pi_{t,j}$$

[4] This is due to the transitivity of edge-connectivity. Specifically, any vertices u, w that have k edge-disjoint paths connecting to the root r also have k edge-disjoint paths between themselves.

In this way, our new concept makes it a relatively simple task to generalize a DP for connectivity-1 problems to a DP for connectivity-k problems (and facilitate the proof of correctness). There is a slight change in the way DPs are designed for each problem, but they follow the same principle.

Organization: We develop our techniques over several sections, and along the way, show non-trivial applications for various SNDP problems. Section 2 provides some notation and important definitions. Section 3 presents the new viewpoint for designing DP and presents a simple showcase by deriving (known) results. Section 4 presents algorithms for EC-SNDP. Lastly, Section 5 presents an approximation algorithm for group connectivity problems.

2 Preliminaries

Tree decomposition: Let G be any graph. A *tree decomposition* of G is a tree $\mathcal{T}$ with a collection of *bags* $\{X_t\}_{t\in V(\mathcal{T})} \subseteq 2^{V(G)}$ (i.e., each node of $\mathcal{T}$ is associated with a subset of nodes of $V(G)$) that satisfies the following properties:

- $V(G) = \bigcup_{t\in V(\mathcal{T})} X_t$
- For any edge $uv \in E(G)$, there is a bag X_t such that $u, v \in X_t$.
- For each vertex $v \in V(G)$, the collection of nodes t whose bags X_t contain v induces a connected subgraph of $\mathcal{T}$. That is, $\mathcal{T}[\{t \in V(\mathcal{T}) : v \in X_t\}]$ is a subtree of $\mathcal{T}$.

The treewidth of G, denoted $tw(G)$, is the minimum integer k for which there exists a tree decomposition $(\mathcal{T}, \{X_t\}_{t\in V(\mathcal{T})})$ such that $\max |X_t| \leq k + 1$ ($\max |X_t| - 1$ is the *width* of $\mathcal{T}$).

Fix a tree decomposition $(\mathcal{T}, \{X_t\})$ with the stated properties. For each node $t \in V(\mathcal{T})$, denote by $\mathcal{T}_t$ the subtree of $\mathcal{T}$ rooted at t and by $p(t)$ the parent node of t in $V(\mathcal{T})$. We also define G_t as the subgraph induced by $\mathcal{T}_t$; that is, $G_t = G\left[\bigcup_{t'\in\mathcal{T}_t} X_{t'}\right]$.

For each $v \in V$, let t_v denote the topmost node for which $v \in X_{t_v}$. For each $t \in V(\mathcal{T})$, we say that an edge $uv \in E(G)$ appears in the bag X_t if $u, v \in X_t$, and *only* if t is the topmost node in which this happens. We denote the edges inside the bag X_t by E_t. For a subset $\mathcal{S} \subseteq V(\mathcal{T})$, we define $X_{\mathcal{S}} := \bigcup_{t\in\mathcal{S}} X_t$.

We will use the following result, which shows that a tree decomposition of G of width $O(tw(G))$ is computable in time $O(2^{O(tw(G))}n)$.

▶ **Theorem 5** ([5]). *There is an algorithm that, given a graph G, runs in time $O(2^{O(tw(G))}n)$ and finds the tree decomposition $(\mathcal{T}, \{X_t\}_{t\in V(\mathcal{T})})$ such that $|X_t| \leq 5tw(G)$ for all t.*

In order to simplify notation, we assume that $\mathcal{T}$ is a binary tree. Furthermore, we require the height of $\mathcal{T}$ to be $O(\log n)$ in Section 5. The following lemma, based on the results of Bodlaender [4], summarizes the properties we assume.

▶ **Lemma 6** (in [12], based on [4]). *Given a tree decomposition $(\mathcal{T}', \{X'_t\}_{t\in V(\mathcal{T}')})$, we can transform it into a tree decomposition $(\mathcal{T}, \{X_t\}_{t\in V(\mathcal{T})})$ with the following properties:*

(i) *the height of $\mathcal{T}$ is at most $O(\log n)$;*
(ii) *each bag X_t satisfies $|X_t| \leq O(w)$;*
(iii) *every leaf bag has no edges ($E_t = \emptyset$ for leaf $t \in \mathcal{T}$);*
(iv) *every non-leaf has exactly 2 children.*

Furthermore, this transformation runs in linear time.

Connection sets and operators: Let $S \subseteq V$. A connection set Λ over S is a subset of $S \times S$ which will be used to list pairs that are connected via a path, i.e., $(u, v) \in \Lambda$ if there is a path connecting u to v.

Let Λ be a connection set over S. The *transitive closure* operator, denoted by $\mathrm{tc}(\cdot)$, is defined naturally such that $\mathrm{tc}(\Lambda)$ contains all pairs (w, w') for which there is a sequence $(w = w_0, w_1, \ldots, w_q = w')$ and $(w_i, w_{i+1}) \in \Lambda$ for all $i < q$. Let $S' \subseteq S$. The *projection operator* "$|$" is defined such that $\Lambda\big|_{S'} = \Lambda \cap (S' \times S')$.

Given two connection sets Λ_1 of S_1 and Λ_2 of S_2, the union $\Lambda_1 \cup \Lambda_2$ is a connection set over $S_1 \cup S_2$.

3 New Key Concept: Global ⇔ Local Checking for DP

This section introduces the key concept devised for handling all our problems systematically.

High-level intuition: Our DP will try to maintain a pair of local and global information about connectivity in the graph. Roughly speaking, a local connection set Γ_t for t (more precisely, for X_t) gives information about connectivity of the solution inside the subgraph G_t (the subgraph induced in subtree $\mathcal{T}_t$), while the other connection Δ_t for t gives information about connectivity of the global solution (i.e., the solution for the whole graph G).

For instance, if we have a tentative solution $Y \subseteq E(G)$, we would like to have the information about the reachability of Y inside each bag, i.e., $\Delta_t = \mathrm{tc}(Y)\big|_t$, so we could check the reachability between u and v simply by looking at whether $(u, v) \in \Delta_t$. However, a DP that is executing at node t may not have this global information, and this often leads to complicated rules to handle this situation.

We observe that global information can be passed along to all cells in the DP with simple local rules so that checking whether the connectivity requirements are satisfied can be done locally inside each DP cell. In the next section, we elaborate on this more formally.

Equivalence: One could imagine having a DP cell $c[t, \Gamma_t, \Delta_t]$ for all possible connection sets Γ_t, Δ_t, which makes a decision on Y_t, the set of edges bought by the solution when executing the DP. The roles of Γ_t and Δ_t are to give information about local and global reachability, respectively. We are seeking a solution Y that is "consistent" with these profiles, where Y can be partitioned based on the tree $\mathcal{T}$ as $Y_t = Y \cap E_t$.

Given Y, we say that the pairs $\{(\Gamma_t, \Delta_t)\}_{t \in V(\mathcal{T})}$ satisfy the *local* (resp., *global*) *connectivity definition* if, for every node $t \in V(\mathcal{T})$ (having left and right children as t' and t'', respectively),

Local

$$\Gamma_t := \begin{cases} \emptyset & \text{if } t \text{ is a leaf of } \mathcal{T} \\ \mathrm{tc}\left(\Gamma_{t'} \cup \Gamma_{t''} \cup Y_t\right)\big|_t & \text{otherwise} \end{cases}$$

$$\Delta_t := \begin{cases} \Gamma_t & \text{if } t = \mathrm{root}(\mathcal{T}) \\ \mathrm{tc}(\Delta_{p(t)} \cup \Gamma_t)\big|_t & \text{otherwise} \end{cases}$$

Global

$$\Gamma_t := \mathrm{tc}\left(Y \cap E(G_t)\right)\big|_t$$

$$\Delta_t := \mathrm{tc}\left(Y\right)\big|_t$$

where the projection operator on X_t is simplified as $|_t$.

The main idea is that the local connectivity definition gives us "local" rules that enforce consistency of consecutive bags, and this would be suitable for being embedded into a DP. The global connectivity rules, however, are not easily encoded into DP, but it is easy to argue intuitively and formally about their properties. The following lemma (proof in Appendix A) shows that the local and global connectivity definitions are, in fact, equivalent.

▶ **Lemma 7.** *Let $Y_t \subseteq E_t$ be a subset of edges and (Γ_t, Δ_t) a pair of connectivity sets for every $t \in V(\mathcal{T})$. Then, the pairs (Γ_t, Δ_t) satisfy the local connectivity definition iff they satisfy the global connectivity definition.*

The notions of local and global connectivity, as well as the equivalence between them can be generalized both for the edge-connectivity version with vertex-costs as well as vertex-connectivity with vertex-costs. We defer the details of this generalization to the full version of the paper.

A warmup application: Steiner trees. We now show an approach that allows us to solve the Steiner Tree problem exactly in $nw^{O(w)}$ time, given a tree decomposition of width w. We remark that the best known algorithm due to Cygan et al. [18] runs in time $2^{O(w)}n$. In this problem, we are given graph $G = (V, E)$ with edge-costs and terminals $T = \{v_1, \ldots, v_h\}$, and the goal is to find a min-cost subset $E^* \subseteq E$ that connects all the terminals. For simplicity, we denote v_1 by "root" r, and add r to every bag. The goal is now to connect the root to all other terminals in $T \setminus r$.

Our DP table has a cell $c[t, \Gamma, \Delta]$ for every node $t \in V(\mathcal{T})$ and every pair of connection sets (Γ, Δ) for t. We initialize the DP table by setting $c[t, \Gamma, \Delta]$ for all the leaf nodes, and setting certain cells as invalid (by setting $c[t, \Gamma, \Delta] = \infty$):

- For every leaf node t and every pair (Γ, Δ), we set $c[t, \Gamma, \Delta] = 0$ if $\Gamma = \emptyset$ and $c[t, \Gamma, \Delta] = \infty$ otherwise.
- We mark the cells $(\text{root}(\mathcal{T}), \Gamma, \Delta)$ as invalid if $\Gamma \neq \Delta$.
- Let $v_i \in T$ be one of the terminals, and $t \in V(\mathcal{T})$ a node. We mark a cell (t, Γ, Δ) as invalid if $v_i \in X_t$ but $(r, v_i) \notin \Delta$.

For all other cells, we compute the result from their children. Let t be a node with left-child t' and right-child t''. Let $(\Gamma, \Delta), (\Gamma', \Delta'), (\Gamma'', \Delta'')$ be pairs of connection sets for t, t', t'', respectively, and $Y_t \subseteq E_t$. We say that (Γ, Δ) is consistent with $((\Gamma', \Delta'), (\Gamma'', \Delta''))$ via Y_t (abbreviated by the notation $(\Gamma, \Delta) \xleftrightarrow{Y_t} ((\Gamma', \Delta'), (\Gamma'', \Delta''))$ if

$$\Gamma = \text{tc}(\Gamma' \cup \Gamma'' \cup Y_t)\big|_t \qquad \Delta' = \text{tc}(\Delta \cup \Gamma')\big|_{t'} \qquad \Delta'' = \text{tc}(\Delta \cup \Gamma'')\big|_{t''}$$

Now, for any choice of valid DP cells (t, Γ_t, Δ_t) and edge subsets $Y_t \subseteq E_t$ for every $t \in V(\mathcal{T})$ (notice that a DP solution uses precisely one cell per node t), we apply Lemma 7 to conclude that since the local connectivity definition is satisfied for (Γ_t, Δ_t) pairs, so does the global connectivity definition. Since, for every valid DP cell (t, Γ, Δ) such that t contains a terminal v_i, $(r, v_i) \in \Delta$, we conclude that every terminal is connected to the root in the solution $Y = \bigcup_{t \in V(\mathcal{T})} Y_t$. Thus, the cost of the optimum solution can be found at one of the cells $c[\text{root}(\mathcal{T}), \Gamma, \Delta]$ (with $\Gamma = \Delta$).

Conversely, given a solution $F \subseteq E(G)$, we can define $F_t := F \cap E_t$, and a pair (Γ_t, Δ_t) for every $t \in V(\mathcal{T})$, using the global connectivity definition. Lemma 7 implies that the pairs (Γ_t, Δ_t) satisfy the local connectivity definition, and therefore define valid DP cells (notice that for every terminal v_i , F connects v_i to the root, so $(r, v_i) \in \Delta_t$ for every $t \in V(\mathcal{T})$ such that $v_i \in X_t$). We thus establish that for every valid DP solution there is a corresponding feasible solution $F \subseteq E(G)$, and vice-versa.

4 Extension to High Connectivity

This section shows how to apply our framework to problems with high connectivity requirements. We focus on edge-connectivity and leave the case of vertex-connectivity to the full version.

When solving a problem in a high connectivity setting, there may be a requirement of k disjoint paths. In particular, for each demand pair (u, v), there must be $k(u, v)$ disjoint paths in the solution. So we could start naturally with a profile of the form:

$$(\Gamma_{t,1}, \ldots, \Gamma_{t,k})(\Delta_{t,1}, \ldots, \Delta_{t,k})$$

for each node $t \in V(\mathcal{T})$, and enforce the local consistency conditions for each coordinate. Here, each pair $(\Gamma_{t,j}, \Delta_{t,j})$ would correspond to connectivity in some subgraph $H_j \subseteq G$, where H_j serve the j-th path of the demand (u, v).

However, this idea does not work as a different demand pair, say (a, b), might use a path that belongs to different subgraphs H_j as defined above. In other words, the disjoint paths for the demand pair (a, b) might require a different partitioning of the solution set H. Therefore, we need to enumerate all possible ways for the demands to "locally" partition the graph and use them to support all k disjoint paths for each one of them. This requires a more careful local consistency check between the DP cells.

We consider the Rooted EC-SNDP problem with edge-costs as an example to explain how to apply our framework to high-connectivity settings, that is, we will prove Theorem 4. For convenience, we add r to every bag. To avoid confusion, the root of the tree $\mathcal{T}$ will be referred explicitly as root($\mathcal{T}$).

The organization of this section is as follows. In Section 4.1, we explain the setup of the cells of the DP table and a high-level intuition about how the DP works. In Section 4.2 we describe the algorithm, in particular, how to compute the values of the DP table. We leave the discussion of its correctness and running time to Appendix B.2.

4.1 Profiles

As in any standard dynamic programming approach based on tree decomposition, we have a profile for each node t, which tries to solve the subproblem restricted to G_t in some way.

Let $t \in V(\mathcal{T})$. A *connection profile* for t is a k-tuple $\vec{\Gamma} = (\Gamma_1, \Gamma_2, \ldots, \Gamma_k)$ such that $\Gamma_i \subseteq X_t \times X_t$. Let $\mathcal{X}_t$ be the set of all connection profiles for t. A profile Ψ of node t is a collection of pairs of connection profiles $(\vec{\Gamma}, \vec{\Delta})$, i.e., $\Psi \subseteq \mathcal{X}_t \times \mathcal{X}_t$. A partial solution $F \subseteq E(G_t)$ is said to be *consistent* with profile Ψ for t if, for all $(\vec{\Gamma}, \vec{\Delta}) \in \Psi$,

- For each $(u, v) \in \Gamma_i$ and $(a, b) \in \Gamma_j$ for $i \neq j$, there are paths $P_{uv}, P_{ab} \subseteq F$ connecting the respective vertices in G_t such that P_{uv} and P_{ab} are edge-disjoint.
- There is a global solution $F' \supseteq F$ such that, for each $(u, v) \in \Delta_i$ and $(a, b) \in \Delta_j$ for $i \neq j$, there are paths $Q_{uv}, Q_{ab} \subseteq F'$ connecting the respective vertices in G such that P_{uv} and P_{ab} are edge-disjoint.

In other words, a solution consistent with a profile must "implement" all connectivity requirements by $\vec{\Gamma}$ and must be extensible to satisfy $\vec{\Delta}$.

Passing down both local and global requirements in the DP table leads to a clean and simple DP algorithm. Our DP table has a cell $c[t, \Psi]$ for each $t \in V(\mathcal{T})$ and each profile Ψ for t. This cell tentatively stores the optimal cost of a solution consistent with profile Ψ.

4.2 The DP

Valid cells: Some table entries do not correspond to valid solutions of the problem, so we mark them as invalid and remove them from consideration (another way to think about this is that we initialize $c[t, \Psi] = \infty$ for all invalid cells), i.e., the following cells are invalid:

- Any leaf that has non-empty connectivity requirements is invalid. That is, $c[t, \Psi] = 0$ if $\Psi \subseteq \{(\emptyset, \ldots, \emptyset)\} \times \mathcal{X}_t$; otherwise, $c[t, \Psi] = \infty$.

- Any cell that cannot be extended into a feasible solution is invalid. If there is no pair $(\vec{\Gamma}, \vec{\Delta}) \in \Psi$ such that $(r, \gamma_i) \in \Delta_j$ for all $j \in [k_i]$, then there are fewer than k_i edge-disjoint paths between r and γ_i, and therefore the cell is invalid, so $c[t_{\gamma_i}, \Psi] = \infty$.
- The node root$(\mathcal{T})$ together with profile Ψ is an invalid cell if there is a pair $(\vec{\Gamma}, \vec{\Delta}) \in \Psi$ such that $\Gamma_j \neq \Delta_j$ for some j. In this case, we set $c[\text{root}(\mathcal{T}), \Psi] = \infty$.

▶ **Lemma 8.** *For every $t \in \mathcal{T}$, there are at most $\exp(w^{O(wk)})$ many valid cells (t, Ψ).*

We remark that, though the solution can be partitioned differently for each demand, the number of different partitions of a solution still depends on its size. Therefore, once we look at the partitions for each bag X_t, the number of possibilities is bounded by a function of w and k.

DP computation: For all other cells, we compute their values from the values of their children. Let t be a node with left-child t' and right-child t''. Let Ψ, Ψ', Ψ'' be their profiles respectively, and $Y_t \subseteq E_t$. We say that Ψ is consistent with (Ψ', Ψ'') via Y_t (abbreviated by $\Psi \xleftrightarrow{Y_t} (\Psi', \Psi'')$) if the following conditions are satisfied. For each pair $(\vec{\Gamma}, \vec{\Delta}) \in \Psi$, there are $(\vec{\Gamma}', \vec{\Delta}') \in \Psi'$ and $(\vec{\Gamma}'', \vec{\Delta}'') \in \Psi''$, together with a partition of Y_t into $Y_1 \cup Y_2 \cup \ldots \cup Y_k$ such that, for every $j \in [k]$,

$$\Gamma_j = \text{tc}(\Gamma'_j \cup \Gamma''_j \cup Y_j)\big|_t \qquad \Delta'_j = \text{tc}(\Delta_j \cup \Gamma'_j)\big|_{t'} \qquad \Delta''_j = \text{tc}(\Delta_j \cup \Gamma''_j)\big|_{t''}$$

Similarly, for any $(\vec{\Gamma}', \vec{\Delta}') \in \Psi'$ (resp., $(\vec{\Gamma}'', \vec{\Delta}'') \in \Psi''$) there are $(\vec{\Gamma}'', \vec{\Delta}'') \in \Psi''$ (resp., $(\vec{\Gamma}', \vec{\Delta}') \in \Psi'$) plus $(\vec{\Gamma}, \vec{\Delta}) \in \Psi$ and some partition of Y_t, satisfying similar conditions as above.

Then the value of $c[t, \Psi]$ can be defined recursively among valid cells:

$$c[t, \Psi] = \min_{\Psi \xleftrightarrow{Y_t} (\Psi', \Psi'')} \left(c[t', \Psi'] + c[t'', \Psi''] + c(Y_t)\right)$$

The final solution can be computed as $\min\left\{c[\text{root}(\mathcal{T}), \Psi] \mid (\forall (\vec{\Gamma}, \vec{\Delta}) \in \Psi)(\forall j \in [k])\Gamma_j = \Delta_j\right\}$. The correctness of this DP is deferred to Appendix B.1.

5 Algorithms for Restricted Group SNDP

In Section 4, we showed how to find the optimum solution to the EC-SNDP by solving a DP. This is only possible because we know the exact demands that correspond to a subproblem, and we mark cells as invalid if these demands are not met. In the Restricted Group SNDP, each group can be connected in subproblems corresponding to different part of the trees, and hence it is no longer possible to solve the DP in a standard way.

We do, however, have sufficient technical tools to prove Theorem 2. Instead of simply solving the DP, we turn the DP table into a tree instance of a variant of GST and, in a second step, apply randomized rounding to obtain a polylogarithmic approximation to the problem. This corresponds to the process of finding a solution in the DP, with the additional constraint that all the demands are met.

Tree Instance: We start by showing how to transform the DP table into a tree $\widetilde{\mathcal{T}}$, where we can solve a variant of the group Steiner tree problem. The following theorem formalizes this transformation, and we dedicate the rest of this section to proving the theorem. For convenience, we add a dummy node t_S with $X_{t_S} = \emptyset$ as the parent of the root node.

▶ **Theorem 9.** *Given a graph G rooted at r with treewidth w and groups $S_i \subseteq V$, there is a tree $\widetilde{\mathcal{T}}$ with groups $\widetilde{S}_i$ and a set of accepted solutions $\widetilde{X}$ such that:*

- **(i)** *the size of $\widetilde{\mathcal{T}}$ is $n^{w^{O(wk)}}$;*
- **(ii)** *for every $F \subseteq E(G)$, there is $X \in \widetilde{X}$ (and vice-versa) such that $c(F) = c(X)$ and, for every $i \in [h]$, F k_i-connects r to $v \in S_i$ iff X connects* $\mathrm{root}(\widetilde{T})$ *to $\tilde{t} \in \widetilde{S}_i$.*

For each cell of the DP table introduced in Section 4.2, we create a node in $\widetilde{\mathcal{T}}$. Namely, we create a vertex $\tilde{t}[t, \Psi]$ for every node $t \in V(\mathcal{T})$ and $\Psi \subseteq \mathcal{X}_t \times \mathcal{X}_t$. The root of the tree is $\tilde{t}[t_S, \{(\emptyset^k, \emptyset^k)\}]$ (this is the only connection profile for t_S). For a node $t \in \mathcal{T}$ with children t', t'', we add connecting nodes $\tilde{t}_c[t, \Psi, \Psi', \Psi'', Y_t]$ connected to the nodes $\tilde{t}[t, \Psi]$, $\tilde{t}[t', \Psi']$, $\tilde{t}[t'', \Psi'']$, for every $Y_t \subseteq E_t$, if $\Psi \xleftrightarrow{Y_t} (\Psi', \Psi'')$. If there is only one child, the connecting node has degree 2, and we consider that $\Psi'' = \{(\emptyset^k, \emptyset^k)\}$ for the purpose of describing the algorithm. An edge from $\tilde{t}[t, \Psi]$ to $\tilde{t}_c[t, \Psi, \Psi', \Psi'', Y_t]$ is labeled with the set of edges Y_t and is assigned cost $c(Y_t)$. All other edges in the instance have cost 0.

Notice that, at this point, $\widetilde{\mathcal{T}}$ is not a tree, but we can turn it into one by making copies of the nodes as required. Specifically, we process the tree in a bottom-up fashion: for each node $\tilde{t}$, we make the same number of copies of $\tilde{t}$ and its descendants as there are incoming edges of $\tilde{t}$ such that each edge is incident to a different copy. In this manner, all the copies of $\tilde{t}$ are now the roots of subtrees, which are disjoint.

For convenience, we denote by $\tilde{t}[t, \Psi]$ (resp., $\tilde{t}_c[t, \Psi, \Psi', \Psi'', Y_t]$) any copy of the original node; when we need to distinguish copies, we denote by $\mathrm{copies}(\tilde{t})$ the set of all copies of a node $\tilde{t}$.

The final step in our construction is to prune the tree by removing nodes that cannot be reached or that represent choices that cannot be part of a feasible solution. To do that, it is sufficient to apply the following rules to exhaustion:

- **(i)** remove $\tilde{t} \in \widetilde{\mathcal{T}}$ if it is not connected to the root;
- **(ii)** remove a connecting node if one of its children was removed;
- **(iii)** remove $\tilde{t}[t, \Psi]$ if it is a leaf node but $\Psi \not\subseteq \{(\emptyset, \dots, \emptyset)\} \times \mathcal{X}_t$ (i.e. $\vec{\Gamma} \neq (\emptyset, \dots, \emptyset)$ for some $(\vec{\Gamma}, \vec{\Delta}) \in \Psi$).

We can now restate the goal of the problem in terms of $\widetilde{\mathcal{T}}$: we want to find nodes $\tilde{t}[t, \Psi_t]$ and edge sets $Y_t \subseteq E_t$ for every node $t \in V(\mathcal{T})$, such that $(\Psi_{t'}, \Psi_{t''}) \xleftrightarrow{Y_t} \Psi_t$ for all non-leaf node t with children t', t''. Further, for every group S_i, there must be a vertex $\gamma_i \in S_i$ and a partition of $Y := \bigcup_{t \in V(\mathcal{T})} Y_t$ into k_i sets, such that each contains a path from r to γ_i.

The set of nodes $\{\tilde{t}[t, \Psi_t]\}_{t \in V(\mathcal{T})}$ with the respective connecting nodes $\tilde{t}[t, \Psi_t, \Psi_{t'}, \Psi_{t''}, Y_t]$ (for all non-leaf nodes $t \in V(\mathcal{T})$) induces a tree $\widetilde{T}$ in $\widetilde{\mathcal{T}}$. We say that such a tree $\widetilde{T}$ is *valid* if every node $\tilde{t}[t, \Psi] \in V(\widetilde{T})$ has exactly one child in the graph (or none if t is a leaf), and every connecting node $\tilde{t}_c[t, \Psi, \Psi', \Psi'', Y_t]$ in $\widetilde{T}$ has full-degree, i.e., all its neighbors are in the solution as well.

For every group S_i, we define $\widetilde{S}_i$ as follows: for every $v \in S_i$ and every $\Psi \in \mathcal{X}_t \times \mathcal{X}_t$, every element of $\mathrm{copies}(\tilde{t}[t_v, \Psi])$ is in $\widetilde{S}_i$ if there is $(\vec{\Gamma}, \vec{\Delta}) \in \Psi$ such that $(r, v) \in \Delta_j$ for all $j \in [k_i]$.

The size of the instance follows by considering its height and maximum degree: the maximum degree of $\widetilde{\mathcal{T}}$ is $\exp(w^{O(wk)})$ by Lemma 8, and there is a tree decomposition of height $O(\log n)$ by Lemma 6, which implies that $\mathrm{height}(\widetilde{\mathcal{T}}) = O(\log n)$. We conclude that $|\widetilde{\mathcal{T}}| = n^{w^{O(wk)}}$.

The correctness of the reduction follows from the correctness of the DP for Rooted EC-SNDP, and its proof is left to Appendix B.1.

Algorithm: We now show how to obtain a valid tree $\widetilde{T}$, given $\widetilde{\mathcal{T}}$. Let T^* be the min-cost valid tree in $\widetilde{\mathcal{T}}$ that connects all the groups $\{\widetilde{S}_i\}_{i\in[h]}$. Chalermsook et al. [12] showed that it is possible to find a valid tree $\widetilde{T}$ with expected cost $c(T^*)$, but whose probability of covering a group is just $O(1/\text{height}(\widetilde{\mathcal{T}}))$.

Using this result, we can obtain valid trees $\widetilde{T}_1, \ldots, \widetilde{T}_\ell$, where $\ell = O(\log n \log h)$. We can then obtain solutions F_j that k-connect the same groups and have the same cost as $\widetilde{T}_j$, for all $j \in [\ell]$, and finally output the solution $F := \bigcup_{j\in[\ell]} F_j$. Since the expected cost of each $\tilde{T}_i$ is $c(T^*)$, the expected cost of F is $O(\log n \log h)c(T^*)$.

By sampling $c \log n \log h$ independent valid trees, for large enough c, we ensure that each group is covered with probability $(1/\log n)^\ell = 1/h^c$, and thus all the groups are covered with high probability (by union bound). For any group that is not covered, we can add minimum-cost edge-disjoint paths from the root (by reduction to min-cost flow) and hence ensure that every group is covered, without increasing the expected cost of the solution. We conclude that the algorithm outputs a randomized $O(\log n \log h)$-approximation to the problem, with high probability.

References

1 MohammadHossein Bateni, Mohammad Taghi Hajiaghayi, and Dániel Marx. Approximation schemes for steiner forest on planar graphs and graphs of bounded treewidth. *J. ACM*, 58(5):21:1–21:37, 2011. doi:10.1145/2027216.2027219.

2 Mihir Bellare, Shafi Goldwasser, Carsten Lund, and A. Russeli. Efficient probabilistically checkable proofs and applications to approximations. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing, May 16-18, 1993, San Diego, CA, USA*, pages 294–304, 1993. doi:10.1145/167088.167174.

3 André Berger and Michelangelo Grigni. Minimum weight 2-edge-connected spanning subgraphs in planar graphs. In *Automata, Languages and Programming, 34th International Colloquium, ICALP 2007, Wroclaw, Poland, July 9-13, 2007, Proceedings*, pages 90–101, 2007.

4 Hans L. Bodlaender. NC-algorithms for graphs with small treewidth. In Jan van Leeuwen, editor, *Graph-Theoretic Concepts in Computer Science, 14th International Workshop, WG '88, Amsterdam, The Netherlands, June 15-17, 1988, Proceedings*, volume 344 of *Lecture Notes in Computer Science*, pages 1–10. Springer, 1988. doi:10.1007/3-540-50728-0.

5 Hans L. Bodlaender, Pål Grønås Drange, Markus S. Dregi, Fedor V. Fomin, Daniel Lokshtanov, and Michal Pilipczuk. A c^k n 5-approximation algorithm for treewidth. *SIAM J. Comput.*, 45(2):317–378, 2016.

6 Richard B. Borie, R. Gary Parker, and Craig A. Tovey. Automatic generation of linear-time algorithms from predicate calculus descriptions of problems on recursively constructed graph families. *Algorithmica*, 7(5&6):555–581, 1992. doi:10.1007/BF01758777.

7 Glencora Borradaile, Erik D. Demaine, and Siamak Tazari. Polynomial-time approximation schemes for subset-connectivity problems in bounded-genus graphs. *Algorithmica*, 68(2):287–311, 2014.

8 Glencora Borradaile, Philip N. Klein, and Claire Mathieu. An $O(n \log n)$ approximation scheme for steiner tree in planar graphs. *ACM Trans. Algorithms*, 5(3):31:1–31:31, 2009. doi:10.1145/1541885.1541892.

9 Glencora Borradaile, Philip N. Klein, and Claire Mathieu. A polynomial-time approximation scheme for euclidean steiner forest. *ACM Trans. Algorithms*, 11(3):19:1–19:20, 2015.

10 Glencora Borradaile and Baigong Zheng. A PTAS for three-edge-connected survivable network design in planar graphs. In *Approximation, Randomization, and Combinatorial*

Optimization. Algorithms and Techniques, APPROX/RANDOM 2017, August 16-18, 2017, Berkeley, CA, USA, pages 3:1–3:13, 2017.

11 İSMET Esra Büyüktahtakin. *Dynamic Programming Via Linear Programming.* John Wiley & Sons, Inc., 2010.

12 Parinya Chalermsook, Syamantak Das, Bundit Laekhanukit, and Daniel Vaz. Beyond metric embedding: Approximating group steiner trees on bounded treewidth graphs. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 737–751, 2017.

13 Parinya Chalermsook, Fabrizio Grandoni, and Bundit Laekhanukit. On survivable set connectivity. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 25–36, 2015.

14 Joseph Cheriyan and Adrian Vetta. Approximation algorithms for network design with metric costs. *SIAM J. Discrete Math.*, 21(3):612–636, 2007. doi:10.1137/040621806.

15 Julia Chuzhoy and Sanjeev Khanna. Algorithms for single-source vertex connectivity. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 105–114, 2008.

16 Julia Chuzhoy and Sanjeev Khanna. An o(k^3log n)-approximation algorithm for vertex-connectivity survivable network design. *Theory of Computing*, 8(1):401–413, 2012. doi:10.4086/toc.2012.v008a018.

17 Bruno Courcelle. The monadic second-order logic of graphs. i. recognizable sets of finite graphs. *Inf. Comput.*, 85(1):12–75, 1990. doi:10.1016/0890-5401(90)90043-H.

18 Marek Cygan, Jesper Nederlof, Marcin Pilipczuk, Michal Pilipczuk, Johan M. M. van Rooij, and Jakub Onufry Wojtaszczyk. Solving connectivity problems parameterized by treewidth in single exponential time. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 150–159, 2011.

19 Artur Czumaj, Michelangelo Grigni, Papa Sissokho, and Hairong Zhao. Approximation schemes for minimum 2-edge-connected and biconnected subgraphs in planar graphs. In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2004, New Orleans, Louisiana, USA, January 11-14, 2004*, pages 496–505, 2004.

20 Daniela Pucci de Farias and Benjamin Van Roy. Approximate dynamic programming via linear programming. In *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada]*, pages 689–695, 2001. URL: http://papers.nips.cc/paper/2129-approximate-dynamic-programming-via-linear-programming.

21 F d'Epenoux. A probabilistic production and inventory problem. *Management Science*, 10(1):98–108, 1963.

22 Naveen Garg, Goran Konjevod, and R. Ravi. A polylogarithmic approximation algorithm for the group steiner tree problem. *J. Algorithms*, 37(1):66–84, 2000.

23 Anupam Gupta, Ravishankar Krishnaswamy, and R. Ravi. Tree embeddings for two-edge-connected network design. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, pages 1521–1538, 2010. doi:10.1137/1.9781611973075.124.

24 Anupam Gupta, Kunal Talwar, and David Witmer. Sparsest cut on bounded treewidth graphs: algorithms and hardness results. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 281–290, 2013. doi:10.1145/2488608.2488644.

25 Eran Halperin and Robert Krauthgamer. Polylogarithmic inapproximability. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing, June 9-11, 2003, San Diego, CA, USA*, pages 585–594, 2003.

26 Kamal Jain. A factor 2 approximation algorithm for the generalized steiner network problem. *Combinatorica*, 21(1):39–60, 2001.

27 Rohit Khandekar, Guy Kortsarz, and Zeev Nutov. Approximating fault-tolerant group-steiner problems. *Theor. Comput. Sci.*, 416:55–64, 2012. doi:10.1016/j.tcs.2011.08.021.

28 Bundit Laekhanukit. An improved approximation algorithm for the minimum cost subset k-connected subgraph problem. *Algorithmica*, 72(3):714–733, 2015.

29 Alan S Manne. Linear programming and sequential decisions. *Management Science*, 6(3):259–267, 1960.

30 R. Kipp Martin, Ronald L. Rardin, and Brian A. Campbell. Polyhedral characterization of discrete dynamic programming. *Operations Research*, 38(1):127–138, 1990. doi:10.1287/opre.38.1.127.

31 Zeev Nutov. Approximating minimum-cost connectivity problems via uncrossable bifamilies. *ACM Trans. Algorithms*, 9(1):1:1–1:16, 2012.

32 Marcin Pilipczuk, Michal Pilipczuk, Piotr Sankowski, and Erik Jan van Leeuwen. Subexponential-time parameterized algorithm for steiner tree on planar graphs. In *30th International Symposium on Theoretical Aspects of Computer Science, STACS 2013, February 27 - March 2, 2013, Kiel, Germany*, pages 353–364, 2013.

33 David P. Williamson, Michel X. Goemans, Milena Mihail, and Vijay V. Vazirani. A primal-dual approximation algorithm for generalized steiner network problems. *Combinatorica*, 15(3):435–454, 1995.

A Details of the Global ⇔ Local Checking for DP

In this section, we will prove a generalized version of Lemma 7, that works for edge-connectivity and vertex-connectivity, both with edge and vertex costs. In vertex-connectivity problems, we are interested in finding internally disjoint paths. In order to handle this setting, we introduce a modified version of transitive closure.

For a set of vertices Z and a set of edges S, we denote by $\mathrm{tc}^*_Z(S)$ the set of all pairs (u,v) such that there is a u-v-path in the graph $(Z \cup \{u,v\}, S)$, that is, a path whose internal vertices are in Z, and whose edges are in S. Formally,

$$\mathrm{tc}^*_Z(S) = \left\{(u,v) \;\middle|\; \exists w_1,\ldots,w_\ell \in Z, \forall i \in [\ell-1], (u,w_1),(w_\ell,v),(w_i,w_{i+1}) \in S\right\}$$

We then keep track, for every node, of which vertices in the bag are allowed to be used in the solution, that is, we use triples (Z,Γ^*,Δ^*), instead of the previously used pairs (Γ,Δ).

Let $W_t \subseteq X_t \setminus X_{p(t)}$ for every $t \in V(\mathcal{T})$, $W = \bigcup_{t\in V(\mathcal{T})} W_t$, $Y_t \subseteq E_t$ for every $t \in V(\mathcal{T})$, $Y = \bigcup_{t\in V(\mathcal{T})} Y_t$, and a triple $(Z_t,\Gamma^*_t,\Delta^*_t)$ for every $t \in V(\mathcal{T})$. For the purposes of this section, we introduce the following definitions for local and global connectivity.

We say that the triples $(Z_t,\Gamma^*_t,\Delta^*_t)$ satisfy the *local (resp. global) connectivity definition* if, for every node $t \in V(\mathcal{T})$,

Local

$$Z_t := \begin{cases} W_t & \text{if } t = \mathrm{root}(\mathcal{T}) \\ \left(Z_{p(t)} \cup W_t\right) \cap V_t & \text{otherwise} \end{cases}$$

$$\Gamma^*_t := \begin{cases} \emptyset & \text{if } t \text{ is a leaf node} \\ \mathrm{tc}^*_{Z_t}\left(\Gamma^*_{t'} \cup \Gamma^*_{t''} \cup Y_t\right)\big|_t & \text{otherwise} \end{cases}$$

$$\Delta^*_t := \begin{cases} \Gamma^*_t & \text{if } t = \mathrm{root}(\mathcal{T}) \\ \mathrm{tc}^*_{Z_t}\left(\Delta^*_{p(t)} \cup \Gamma^*_t\right)\big|_t & \text{otherwise} \end{cases}$$

Global

$$\begin{aligned} Z_t &:= W \cap X_t \\ \Gamma_t^* &:= \mathrm{tc}_W^*\left(Y \cap E(G_t)\right)\big|_t \\ \Delta_t^* &:= \mathrm{tc}_W^*\left(Y\right)\big|_t \end{aligned}$$

We then prove the following lemma, proving that the given local and global connectivity definitions are equivalent.

▶ **Lemma 10.** *Let $W_t \subseteq X_t \setminus X_{p(t)}$ be a subset of vertices, $Y_t \subseteq E_t$ a subset of edges and $(Z, \Gamma_t^*, \Delta_t^*)$ a triple of profiles for every $t \in V(\mathcal{T})$.*

Then, the triples $(Z_t, \Gamma_t^*, \Delta_t^*)$ satisfy the local connectivity definition iff they satisfy the global connectivity definition.

Before proving the lemma, we show how Lemma 7 follows. We will prove that, if we fix $W_t = X_t \setminus X_{p(t)}$ and $Z_t = X_t$, the definitions of Lemmas 7 and 10 are equivalent.

For this, it is sufficient to see that $\mathrm{tc}_W^*(S) = \mathrm{tc}_{V(G)}^*(S) = \mathrm{tc}(S)$, and that the common vertices in $\Gamma_{t'}^*$, $\Gamma_{t''}^*$, and Y_t are all in X_t, thus

$$\mathrm{tc}_{Z_t}^*\left(\Gamma_{t'}^* \cup \Gamma_{t''}^* \cup Y_t\right)\big|_t = \mathrm{tc}\left(\Gamma_{t'}^* \cup \Gamma_{t''}^* \cup Y_t\right)\big|_t$$

Similarly, since $\Delta_{p(t)}^*$ and Γ_t^* only intersect inside $X_t \times X_t$,

$$\mathrm{tc}_{Z_t}^*\left(\Delta_{p(t)}^* \cup \Gamma_t^*\right)\big|_t = \mathrm{tc}\left(\Delta_{p(t)}^* \cup \Gamma_t^*\right)\big|_t$$

We conclude that when $Z_t = X_t$, $\Gamma_t^* = \Gamma_t$ and $\Delta_t^* = \Delta_t$, the proof follows.

The following technical lemma will be useful when proving Lemma 10.

▶ **Lemma 11** (Path Lemma)**.** *Let G be any graph and $\mathcal{T}$ be a tree decomposition of G. Let $t \in V(\mathcal{T})$ and P be a path of length at least 2 whose endpoints x, y are the only vertices of P in t, that is, $V(P) \cap X_t \subseteq \{x, y\}$.*

Then there is a connected (subtree) component $\mathcal{T}'$ in $\mathcal{T} \setminus t$ such that, for any edge $ab \in E(P)$, $\mathcal{T}'$ has a node t' that contains ab, i.e., every edge $ab \in E_{t'}$ for some node $t' \in V(\mathcal{T}')$.

Proof. We provide a simple proof by contradiction. Assume that there are two consecutive edges, $ab, bc \in E(P)$ that are in different connected components of $\mathcal{T} \setminus t$ (otherwise, all edges must be in the same component). Since the set of nodes whose bags contain b must be connected in $\mathcal{T}$ but is not connected in $\mathcal{T} \setminus \{t\}$, $b \in X_t$, and we reach a contradiction. ◀

Proof of Lemma 10. We remark that the function tc^* shares some properties with the usual definition of transitive closure, which are used throughout the proof:

▶ **Observation 12.** *The function tc^* satisfies the following properties:*

- $\mathrm{tc}_Z^*(\mathrm{tc}_Z^*(Y)) = \mathrm{tc}_Z^*(Y)$
- $\mathrm{tc}_{Z'}^*(Y') \subseteq \mathrm{tc}_Z^*(Y)$ *if* $Z' \subseteq Z$, $Y' \subseteq Y$

Equivalence for Z_t:

We prove that the two definitions for Z are equivalent by induction on the depth of the node. At the root, we have that $W_{\mathrm{root}(\mathcal{T})} = W \cap X_{\mathrm{root}(\mathcal{T})}$, so the equivalence holds.

For the induction step, let $t \in V(\mathcal{T})$ be a node other than the root. Then

$$\begin{aligned}\left(Z_{p(t)} \cup W_t\right) \cap X_t &= \left(Z_{p(t)} \cap X_t\right) \cup (W_t \cap X_t) \\ &\subseteq (W \cap X_t) \cup (W \cap X_t) \\ &= W \cap X_t\end{aligned}$$

The second step follows from the induction hypothesis, as well as the definition of W. We now prove the converse inclusion.

$$\begin{aligned}W \cap X_t &= \left(W \cap X_{p(t)} \cap X_t\right) \cup \left(W \cap \left(X_t \setminus X_{p(t)}\right)\right) \\ &\subseteq \left(Z_{p(t)} \cap X_t\right) \cup W_t \\ &= \left(Z_{p(t)} \cup W_t\right) \cap X_t\end{aligned}$$

We use the induction hypothesis, as well as the fact that $W_t \subseteq X_t$.

Equivalence for Γ^*_t:

We prove the statement by induction on the height of a node t. Since $E(G_t) = \emptyset$, both definitions are equivalent for every leaf t.

Let t be any node. By the induction hypothesis, $\Gamma^*_{t'}, \Gamma^*_{t''} \subseteq \mathrm{tc}^*_W\left(Y \cap E(G_t)\right)$. Therefore,

$$\mathrm{tc}^*_{Z_t}\left(\Gamma^*_{t'} \cup \Gamma^*_{t''} \cup Y_t\right)\big|_t \subseteq \mathrm{tc}^*_W\left(Y \cap E(G_t)\right)\big|_t$$

Here, we use that $Z_t = W \cap X_t \subseteq W$.

To prove the converse inclusion, let $(u, v) \in \mathrm{tc}^*_W\left(Y \cap E(G_t)\right)\big|_t$. By definition, there must be a path p between u and v using internal vertices in W. Let $u = w_0, w_1, \ldots, w_\ell = v$ be all the vertices of p (in the correct order) that are also in X_t.

Each pair (w_i, w_{i+1}) is connected by a subpath of p. By Lemma 11, (w_i, w_{i+1}) is either an edge in Y_t, or the subpath is fully contained in $Y \cap E(G_{t'})$ or $Y \cap E(G_{t''})$, and uses internal vertices in W. In the first case, $(w_i, w_{i+1}) \in Y_t$, while in the remaining cases, (w_i, w_{i+1}) is in $\Gamma_{t'}$ or $\Gamma_{t''}$, by the induction hypothesis. We conclude that, since $w_i \in Z_t = W \cap X_t$, for $i \in [l-1]$, then (u, v) is in

$$\mathrm{tc}^*_{Z_t}\left(\Gamma^*_{t'} \cup \Gamma^*_{t''} \cup Y_t\right)\big|_t$$

Equivalence for Δ^*_t:

We now prove that both definitions for Δ^* are equivalent, by using induction on the depth of the nodes. For the node $\mathrm{root}(\mathcal{T})$, we have $E(G_{\mathrm{root}(\mathcal{T})}) = E(G)$, and thus the base case follows.

For the induction step, we remark that $\Delta^*_{p(t)}, \Gamma^*_t \subseteq \mathrm{tc}^*_W\left(Y\right)$ by the induction hypothesis together with the statement of the lemma for Γ^*. Therefore,

$$\mathrm{tc}^*_{Z_t}\left(\Delta^*_{p(t)} \cup \Gamma^*_t\right)\big|_t \subseteq \mathrm{tc}^*_W\left(Y\right)\big|_t$$

For the reverse inclusion, we fix a pair $(u, v) \in \mathrm{tc}^*_W\left(Y\right)\big|_t$, and a path p that connects u to v in Y using internal vertices in W. Further, let $u = w_0, w_1, \ldots, w_\ell = v$ be all the vertices of p (in the correct order) that are also in X_t.

By Lemma 11, (w_i, w_{i+1}) is either an edge in $Y(b)$, or the subpath p' of p connecting w_i and w_{i+1} is contained either in $Y \cap E(G_{t'})$, $Y \cap E(G_{t''})$ or $Y \cap (E \setminus E(G_t))$. For all but the last case, $(w_i, w_{i+1}) \in \Gamma^*_t$, by definition. In the remaining case, it must be that the

vertices of p' are also contained in $X_{\mathcal{T}\setminus\mathcal{T}_b}$. Specifically, because the nodes whose bags contain a given vertex must form a connected component of $\mathcal{T}$, $w_i, w_{i+1} \in X_{p(t)}$, which implies $(w_i, w_{i+1}) \in \mathrm{tc}^*_W(Y)\big|_{p(t)} = \Delta^*_{p(t)}$.

In any case, since $w_i \in Z_t = W \cap X_t$, for $i \in [l-1]$, we conclude that every pair (w_i, w_{i+1}) and thus (u, v), are contained in $\mathrm{tc}^*_{Z_t}\left(\Delta^*_{p(t)} \cup \Gamma^*_t\right)\big|_t$. ◀

B Details of the DP for EC-SNDP

B.1 Correctness

The following two lemmas imply the correctness of our DP.

▶ **Lemma 13.** *Let $F \subseteq E(G)$ be a feasible solution. Then, for every node t, there is a profile Ψ_t for t such that (t, Ψ_t) is valid; for any node t with children t', t'', then $\Psi \xleftrightarrow{Y_t} (\Psi', \Psi'')$, where $Y_t = F \cap E_t$. Furthermore, $c[\mathrm{root}(\mathcal{T}), \Psi_{\mathrm{root}(\mathcal{T})}] = c(F)$.*

Proof. We first observe that for each demand (γ_i, k_i), $i \in [h]$ the solution F can be partitioned into $F = \bigcup_{j\in[k]} F^i_j$ such that the partition F^i_j contains a path connecting γ_i to the root r. Note that, if $k_i < k$, there might not be any path in the partitions $F_{k_i+1}, \ldots, F_k$.

Let t be a node in $\mathcal{T}$. We define Ψ_t as follows. We define the pair (omitting the script of t for convenience) $(\vec{\Gamma}^i(t), \vec{\Delta}^i(t))$ for all $i \in [h]$ as follows.

$$\Gamma^i_j(t) = \mathrm{tc}(F^i_j \cap E(\mathcal{T}_t))\big|_t$$
$$\Delta^i_j(t) = \mathrm{tc}(F^i_j)\big|_t$$

We now define $\Psi_t = \{(\vec{\Gamma}^i(t), \vec{\Delta}^i(t)) : i \in [h]\}$.

We first show that any cell (t, Ψ_t) is valid. For any leaf node t, $E_t = \emptyset$ and hence $\Gamma^i_j(t) = \emptyset$, $i \in [h]$, $j \in [k]$. For the node $\mathrm{root}(\mathcal{T})$, $E(\mathcal{T}_t) = E$ and hence $\Gamma^i_j(t) = \Delta^i_j(t)$, $i \in [h]$, $j \in [k]$. Finally, since $\mathcal{P}^i_j$ connects r to γ_i for all $i \in [h]$, $j \in [k_i]$, then $(r, \gamma_i) \in \Delta^i_j(t_{\gamma_i})$. We conclude that any cell (t, Ψ_t) defined as above is marked valid.

Finally we define, for any node t, a set of edges $Y(t) \subseteq E_t$ along with a suitable partition of $Y(t)$. Let $Y(t) = F \cap E_t$ and $Y^i_j(t) = F^i_j \cap E_t$. The subsets $Y^i_j(t)$ indeed form a partition owing to the disjointness of the sets F^i_j, for all $j \in [k_i]$ and a fixed $i \in [h]$.

It is clear from the above definitions of Γ^i_j and Δ^i_j that they satisfy the global connectivity conditions for Y^i_j. Applying Lemma 7, the local conditions must also be satisfied for the node t along with its children t', t'' and $Y(t)$. This gives us $\Psi \xleftrightarrow{Y(t)} (\Psi', \Psi'')$ and we are done.

Notice that the $c[\mathrm{root}(\mathcal{T}), \Psi_{\mathrm{root}(\mathcal{T})}] = \sum_{t\in V(\mathcal{T})} c(Y(t)) = c(F)$. ◀

We remark that the DP uses exactly one cell per node in any valid solution.

▶ **Lemma 14.** *Let $\mathcal{C} = \{(t, \Psi_t) : t \in V(\mathcal{T})\}$ be the set of cells selected by the dynamic programming solution. Then there exists a set of edges $F \subseteq E$ such that for each demand (γ_i, k_i), $i \in [h]$, there exist k_i edge-disjoint paths connecting r to γ_i. Furthermore, $c(F) = c[\mathrm{root}(\mathcal{T}), \Psi_{\mathrm{root}(\mathcal{T})}]$.*

Proof. First we define the set F. Consider the cells in $\mathcal{C}$. Since each cell (t, Ψ_t) is picked by the DP, there must exist a set of edges $Y_t \subseteq E_t$ such that for some pair of children cells $\{(t', \Psi_{t'}), (t'', \Psi_{t''})\} \in \mathcal{C}$, $\Psi_t \xleftrightarrow{Y_t} (\Psi_{t'}, \Psi_{t''})$. Define $F = \bigcup_{(t,\Psi_t)\in\mathcal{C}} Y_t$. We prove that for any demand (γ_i, k_i), $i \in [h]$, there exist edge-disjoint paths $\mathcal{P}^i_j$, $j \in [k_i]$ in F that connect r to γ_i.

For a demand vertex γ_i, $i \in [h]$, consider the node $t^\star = t_{\gamma_i}$ and the cell $(t^\star, \Psi_{t^\star}) \in \mathcal{C}$. Since this cell is valid, there exists a connection profile $(\vec{\Gamma}, \vec{\Delta}) \in \Psi_{t^\star}$ such that $(r, \gamma_i) \in \Delta_j$ for all $j \in [k_i]$.

We now suitably define connection profiles $(\vec{\Gamma}^t, \vec{\Delta}^t)$ and sets of edges $Y_{t,j}$ for every other node $t \in V(\mathcal{T})$, and prove that these elements satisfy the local connectivity property of Lemma 7. We explicitly describe the definition for the children nodes t' and t'' of $t^\star$. The definition for every other node in the subtree $\mathcal{T}_t$ can be carried out in a similar recursive fashion.

By definition of consistent DP cells, there exists $(\vec{\Gamma}', \vec{\Delta}') \in \Psi_{t'}$, $(\vec{\Gamma}'', \vec{\Delta}'') \in \Psi_{t''}$ and a partition $Y_{t^\star} = \bigcup_{j \in [k]} Y_{t^\star, j}$ such that, for all $j \in [h]$, the triplet $(\Gamma_j, \Delta_j), (\Gamma'_j, \Delta'_j), (\Gamma''_j, \Delta''_j)$ satisfies the local connectivity conditions for $Y_{t^\star, j}$. We take $(\vec{\Gamma}^{t'}, \vec{\Delta}^{t'}) = (\vec{\Gamma}', \vec{\Delta}')$, $(\vec{\Gamma}^{t''}, \vec{\Delta}^{t''}) = (\vec{\Gamma}'', \vec{\Delta}'')$.

A similar recursive definition works for all nodes in $\mathcal{T} \setminus \mathcal{T}_t$, starting with $(\vec{\Gamma}, \vec{\Delta})$ and defining a suitable connection profile in $p(t)$.

We now apply the equivalence from Lemma 7 to conclude that the global connectivity definition is satisfied by (Γ^t_j, Δ^t_j) and edge set $Y_{t,j}$ for any node $t \in V(\mathcal{T})$. As a consequence, there exist paths connecting r to γ_i in $\bigcup_{t \in V(\mathcal{T})} Y_{t,j}$, for $j \in [k_i]$. Since the sets $Y_{t,j}$ are disjoint for $j \in [k_i]$, we obtain the required k_i edge-disjoint paths in the sets $Y_j = \bigcup_{t \in V(\mathcal{T})} Y_{t,j}$. ◀

B.2 Running Time Analysis

To bound the running time of the DP algorithm, we start by proving a bound on the number of cells in the DP table (Lemma 8), and then show how this implies the running time of the algorithm.

Proof of Lemma 8. The following observations are used to prove the lemma:

▶ **Observation 15.**

1. *The number of possible subsets of edges between w elements is 2^{w^2}.*
2. *The number of possible partitions of such a subset of edges is k^{w^2}.*
3. *The number of possible equivalence relations in a set of w elements is w^w.*

Let $t \in \mathcal{T}$. We know that $\Psi \subseteq \mathcal{X}_t \times \mathcal{X}_t$, whose size can be up to 2^{kw^2}. However, it is sufficient to consider equivalence relations in the node t, as we always take the transitive closure in definitions. Therefore, there are at most w^w possibilities for each Γ_j, Δ_j (Observation 15) and at most $2^{w^{2wk}}$ possibilities for Ψ. ◀

Note that the algorithm itself does one of the two possible options for each cell: it either initializes itself, for which it needs to check every element of Ψ, taking time $O(w^{2wk}kw)$; or it computes the value based on children cells, for which it enumerates all sets Ψ', Ψ'' and checks them for consistency. The number of such sets to check is again $2^{w^{2wk}}$, and checking each triple of sets takes time polynomial in w^{wk} and k^{w^2}. In sum, the algorithm takes time $O(n \exp(w^{O(wk)}))$.

Perturbation Resilient Clustering for k-Center and Related Problems via LP Relaxations

Chandra Chekuri
Department of Computer Science, University of Illinois, Urbana-Champaign, IL 61801, USA
chekuri@illinois.edu

Shalmoli Gupta
Department of Computer Science, University of Illinois, Urbana-Champaign, IL 61801, USA
sgupta49@illinois.edu

Abstract

We consider clustering in the perturbation resilience model that has been studied since the work of Bilu and Linial [16] and Awasthi, Blum and Sheffet [8]. A clustering instance $\mathcal{I}$ is said to be α-perturbation resilient if the optimal solution does not change when the pairwise distances are modified by a factor of α and the perturbed distances satisfy the metric property – this is the metric perturbation resilience property introduced in [4] and a weaker requirement than prior models. We make two high-level contributions.

- We show that the natural LP relaxation of k-center and asymmetric k-center is integral for 2-perturbation resilient instances. We belive that demonstrating the goodness of standard LP relaxations complements existing results [12, 4] that are based on new algorithms designed for the perturbation model.
- We define a simple new model of perturbation resilience for clustering with *outliers*. Using this model we show that the unified MST and dynamic programming based algorithm proposed in [4] exactly solves the clustering with outliers problem for several common center based objectives (like k-center, k-means, k-median) when the instances is 2-perturbation resilient. We further show that a natural LP relxation is integral for 2-perturbation resilient instances of k-center with outliers.

2012 ACM Subject Classification Theory of computation → Facility location and clustering

Keywords and phrases Clustering, Perturbation Resilience, LP Integrality, Outliers, Beyond Worst Case Analysis

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.9

Related Version A full version of the paper is available at `https://arxiv.org/abs/1806.04202`.

Funding Work on this paper supported in part by NSF grants CCF-1319376 and CCF-1526799.

Acknowledgements CC thanks Mohit Singh for initial discussions on the integrality of the LP relaxation for 2-perturbation-resilient instances of k-median. We thank Yury Makarychev for comments on Voronoi clustering for k-center.

1 Introduction

Clustering is an ubiquitous task that finds applications in numerous areas since it is a basic primitive in data analysis. Consequently, clustering methods are extensively studied in many scientific communities and there is a vast literature on this topic. In a typical clustering problem the input is a set of points with a notion of similarity (also called *distance*) between

every pair of points, and a parameter k, which specifies the desired number of clusters. The goal is to partition the points into k clusters such that points assigned to the same cluster are similar. One way to obtain this partition is to select k centers and then assign each point to the nearest center. The quality of the clustering can be measured in terms of an objective function. Some of the popular and commonly studied ones are k-median (sum of distances of points to nearest center), k-means (sum of squared distances of points to nearest center), and k-center (maximum distance between a point to its nearest center). These are center-based objective functions. Unlike some applications in Operations Research, in many clustering problems in data analysis, the objective function is a proxy to identify the clusters and the actual value of the objective function is not necessarily meaningful. Clustering is often considered in the presence of outliers. In this setting the goal is to find the best clustering of the input after removing (at most) a specified number (or fraction) of points – this is useful in practice when the input data is noisy.

Most of the natural optimization problems that arise in clustering turn out to be NP-Hard. Extensive work exists on approximation algorithm design as well as heuristics. Although clustering and its variants are intractable in the worst case, various heuristic based algorithms like Lloyd's, K-Means++ perform very well in practice and are routinely used – at the same time some of these heuristics have poor worst-case approximation performance. On the other hand algorithms designed for worst-case approximation bounds may not work well in practice or may not be sufficiently fast for large data sets. To bridge this gap between theory and practice, there has been an increasing emphasis on *beyond worst case analysis*. Several models have been proposed to understand real-world instances and why they may be computationally easier. One such model is based on the notion of *instance stability*. This is based on the assumption that typical instances have a clear underlying optimal clustering (also known as *ground-truth clustering*) which is significantly better than all other clusterings, and remains the same under small perturbations.

The notion of stability/perturbation resilience was formalized in the work of Bilu and Linial [16] initially for Max Cut, and by Awasthi, Blum and Sheffet [8] for clustering. For clustering problems, an instance $\mathcal{I}$ is said to be α-perturbation resilient for some $\alpha > 1$ if the optimum clustering remains the same even if pairwise distances between points are altered by a multiplicative factor of at most α. Intuitively, α determines the degree of resilience of the instance, with a higher value translating to more structured, and separable instances. In the past few years, there has been increasing interest in understanding stable/perturbation-resilient instances. After several papers [8, 13, 12], a recent result by Angelidakis, Makarychev and Makarychev [4] showed that 2-perturbation resilient instances of several clustering problems with center based objectives (which includes k-median, k-center, k-means) can be solved exactly in polynomial time. For k-center finding the optimum solution for $(2-\delta)$-perturbation resilient instances is NP-Hard [12]. One criticism of perturbation resilience for clustering was the assumption in some earlier works that the optimum clustering remains stable under perturbation of the original metric d even when perturbed distance d' itself may not be a metric. Interestingly the results of [4] hold even under the weaker assumption of *metric* perturbation resilience, which constrains the perturbed pairwise distances to be a metric. The results in [4] are based on a simple and unified algorithm that computes the MST T of the given set of points and then applies dynamic programming on T to find the clusters; it is only in the second step that the specific objective function is used. We believe that empirically evaluating the performance of this algorithm, and related heuristics, on real-world data is an interesting avenue and plan to study it. Our work in this paper is motivated by the existing work and several interrelated questions on theoretical concerns, that we discuss next.

One of the objectives in beyond-worst-case analysis is to explain the empirical success of existing algorithms and mathematical programming formulations. For stable instances of Max-Cut and Minimum Multiway Cut, convex relaxations are known to be integral for various bounds on the perturbation parameter [31, 4]. In the context of k-median and k-means Awasthi et al. [9] showed that if the data is generated uniformly at random from k unit balls with well-separated centers, convex relaxations (linear and semi-definite) give an optimal itegral solution under appropriate separation conditions on the centers. However, for perturbation resilient clustering instances not much is known about the the natural LP relaxations. This raises a natural question.

▶ **Question 1.** *Are the natural LP relaxations for* 2*-metric perturbation resilient instances of clustering problems integral?*

There are several advantages in proving that well-known relaxations are integral. First, they provide evidence of the goodness of the relaxation; often these relaxations also have worst-case approximation bounds. Second, when the relaxation does not give an integral solution for a given instance we can deduce that the instance is *not* perturbation resilient.

As we remarked, one major takeaway from the paper of Angelidakis et al. [4], apart from its strong theoretical results, is the simple and unified algorithm that they propose which may lead to an effective heuristic. In real-world data there is often noise, and it would be useful to develop algorithms in the more general setting of clustering with outliers. This leads us to the question,

▶ **Question 2.** *Is there any stability model under which the algorithm proposed by [4] gives optimal solution for the problem of clustering with outliers?*

We remark that even for instances without outliers, removing a small fraction of the points can lead to a residual instance which has better stability parameters than the initial one. Thus, clustering with outlier removal is relevant even when there is no explicit noise.

1.1 Our Results

In this paper we address the preceding questions and obtain the following results.

- We show that a natural LP relaxation for k-center has an optimum integral solution for 2-metric-perturbation resilience instances[1]. Thus, when running the LP on a clustering instance, either we are guaranteed to have found the optimal solution (if the LP solution is integral), or we are guaranteed that the instance is not 2-perturbation resilient (if the LP solution is not integral). The previous algorithms of Angelidakis et al. [4], and Balcan et al. [12] do not have this guarantee, and could be arbitrarily bad if the instance is not 2-PR.
- Motivated by the work of [12] we consider the *asymmetric* k-center (Asym-k-center) problem. We show that a natural LP relaxation has an optimum integral solution for 2-*metric*-perturbation resilient instances[2]. For Asym-k-center the worst-case integrality gap of the LP relaxation is known to be $\Theta(\log^* k)$ [5, 23]. Previously [12] described a specific combinatorial algorithm that outputs an optimum solution for 2-perturbation resilient instances. We obtain it via the LP relaxation in the weaker metric perturbation model.

[1] For k-center it is known [12, 14] that *any* 2-approximation algorithm yields an optimum solution for 2-perturbation resilient instances. Although the LP lower bound provides a 2-approximation it is not immediate that it would be exact for perturbation-resilience instances

[2] In the asymmetric setting the perturbed distances should satisfy triangle inequality but symmetry is not required.

- We define a simple model of perturbation resilience for clustering with *outliers*. It is a clean extension of the existing perturbation resilience model. We show that under this new model, a modification of the algorithm of Angelidakis et al. [4] gives an exact solution for the outliers problem (for k-median, k-means, k-center and outer ℓ_p based objectives). This algorithm may lead to an interesting heuristic for clustering (noisy) real-world instances. We also show that for a 2-perturbation resilient instance of k-center with outliers, a natural LP relaxation has an optimum integral solution[3].

Our results show the efficacy of LP relaxations for k-center and its variants. We also demonstrate, via a natural model, that the interesting algorithm from [4] extends to handle outliers. Perturbation resilience appears to be a simple definition but it is hard to pin down its precise implications. Prior work demonstrates that observations and algorithms that appear simple in retrospect have not been easy to find. For k-center and Asym-k-center we work with notion of perturbation resilience under Voronoi clusterings as was done in [12]; this is the more restrictive version. See Section 2 for the formal definitions.

We would like to understand the integrality gap of the natural LP relaxations for perturbation resilient instances of k-median and k-means. We believe that the following open question is quite interesting to resolve.

▶ **Question 3.** *Is there a fixed constant α such that the natural LP relaxation for k-median (similarly k-means) has an integral optimum solution for every α-perturbation resilient instance[4]?*

Please see Appendix A for other related work. Most proofs are omitted in this conference version due to space constraints. We encourage the reader to read the full version of the paper that will be available shortly on the ArXiv.

2 Preliminaries

2.1 Definitions & Notations

Clustering. An instance $\mathcal{I}$ of a clustering problem is defined by the tuple (V, d, k), where V is a set of n points, $d : V \times V \to \mathbb{R}_{\geq 0}$ is a metric distance function, and k is an integer parameter. The goal is to find a set of k distinct points $S = \{c_1, \ldots, c_k\} \subseteq V$ called *centers* such that an objective function defined over the points is optimized. The objective function, also known as clustering cost, can be defined in various ways, and depends on the problem in hand. Here, we are interested in the k-median, and k-center objectives. Given a set of centers $S = \{c_1, \ldots, c_k\}$ these objectives are defined as follows: [i] (k-median) $\texttt{cost}_d(S) = \sum_{u \in V} d(S, u)$ [ii] (k-means) $\texttt{cost}_d(S) = \sum_{u \in V} d^2(S, u)$ [iii] (k-center) $\texttt{cost}_d(S) = \max_{u \in V} d(S, u)$; where $d(S, u) = \min_{i \in \{1,\ldots,k\}} d(c_i, u)$.

The Voronoi partition induced by the centers, gives a natural way of clustering the input point set. In fact, the inherent goal of clustering is to uncover the underlying partitioning of points, and one expects with correct choice of distance modeling, "k", and objective function,

[3] It will be interesting to see whether *any* 2-approximation algorithm for k-center-outlier gives an optimum solution for 2-perturbation resilient instances. A starting point would be to check the 2-approximation algorithm in [19].

[4] It may be possible to answer this question in the positive if we additionally assume that the optimum clusters are balanced in terms of number of points. However, we feel that such an assumption does not shed light on the structure of perturbation resilient instances that are not balanced.

the Voronoi partition induced by the optimal set of centers will reveal the underlying clustering. Throughout this paper, whenever we mention *optimal clustering*, we indicate the Voronoi partition corresponding to the optimal set of centers. Thus with this dual view of the clustering problem, given a set of centers $S = \{c_1, \ldots, c_k\}$, and corresponding Voronoi partition $\mathcal{C} = \{C_1, \ldots, C_k\}$, the clustering cost can be rewritten as: [i] (k-median) $\texttt{cost}_d(\mathcal{C}, S) = \sum_{i=1}^{k} \sum_{u \in C_i} d(c_i, u)$ [ii] (k-means) $\texttt{cost}_d(\mathcal{C}, S) = \sum_{i=1}^{k} \sum_{u \in C_i} d^2(c_i, u)$ [iii] (k-center) $\texttt{cost}_d(\mathcal{C}, S) = \max_{i \in \{1,\ldots,k\}} \max_{u \in C_i} d(c_i, u)$.

So far, in the clustering problem instance, we considered the distance function d to be a metric. However, this may not always be the case. Specifically, for the k-center objective, a generalization which is also studied is the Asymmetric k-center problem (Asym-k-center), where the distance function d in the input instance $\mathcal{I} = (V, d, k)$ is an asymmetric distance function. In other words, d obeys triangle inequality, but not symmetry. That is $d(u, v) \leq d(u, w) + d(w, v)$ for all $u, v, w \in V$. However $d(u, v)$ may be not be same as $d(v, u)$. The objective is the k-center objective, but because the distance is assymetric, order matters – we define the cost in terms of distance *from the center to the points* i.e. given a center c and a point u, $d(c, u)$ is used to define cost. To reiterate, given a set of centers $S = \{c_1, \ldots, c_k\}$ and corresponding Voronoi partition (w.r.t $d(c_i, u)$) $\mathcal{C} = \{C_1, \ldots, C_k\}$, the clustering cost is: (Asym-k-center) $\texttt{cost}_d(\mathcal{C}, S) = \max_{i \in \{1,\ldots,k\}} \max_{u \in C_i} d(c_i, u)$.

Clustering with Outliers. An instance $\mathcal{I}$ of a clustering with outliers problem is defined by the tuple (V, d, k, z), where V is a set of n points, $d : V \times V \to \mathbb{R}_{\geq 0}$ is a metric distance function, and k, z are integer parameters. The goal is to identify z points $Z \subseteq V$ as *outliers* and partition the remaining $V \setminus Z$ points into k clusters such that the clustering cost is minimized. Formally, given a set of outliers Z, a set of centers $S = \{c_1, \ldots, c_k\} \subseteq V \setminus Z$, and a Voronoi partition of $V \setminus Z$, $\mathcal{C} = \{C_1, \ldots, C_k\}$ induced by S, the clustering cost is defined as: [i] (k-median-outlier) $\texttt{cost}_d(\mathcal{C}, S; Z) = \sum_{i=1}^{k} \sum_{u \in C_i} d(c_i, u)$ [ii] (k-means-outlier) $\texttt{cost}_d(\mathcal{C}, S; Z) = \sum_{i=1}^{k} \sum_{u \in C_i} d^2(c_i, u)$ [iii] (k-center-outlier) $\texttt{cost}_d(\mathcal{C}, S; Z) = \max_{i \in \{1,\ldots,k\}} \max_{u \in C_i} d(c_i, u)$.

Perturbation Resilience. A clustering instance $\mathcal{I} = (V, d, k)$ is α-metric perturbation resilient (α-PR) for a given objective function, if for any metric [5] distance function $d' : V \times V \to \mathbb{R}_{\geq 0}$, such that for all $u, v \in V$, $\frac{d(u,v)}{\alpha} \leq d'(u, v) \leq d(u, v)$, the unique optimal clustering of $\mathcal{I}' = (V, d', k)$ is identical to the unique optimal clustering of $\mathcal{I}$.

Note that after perturbation the optimal centers may change, however for the instance to be perturbation resilient, the optimal clustering i.e. Voronoi partition induced by the optimal centers must stay the same. Unless otherwise noted, for the rest of the paper α-perturbation resilient indicates metric perturbation resilience.

Outlier Perturbation Resilience. A clustering with outliers instance $\mathcal{I} = (V, d, k, z)$ is α-metric outlier perturbation resilient (α-OPR) for a given objective function, if for any metric distance function $d' : V \times V \to \mathbb{R}_{\geq 0}$, such that for all $u, v \in V$, $\frac{d(u,v)}{\alpha} \leq d'(u, v) \leq d(u, v)$, the unique optimal clustering and outliers of $\mathcal{I}' = (V, d', k, z)$ are identical to the optimal solution of $\mathcal{I}$.

It is easy to see, if a clustering with outliers instance (V, d, k, z) with unique optimal clusters $\mathcal{C}$ and outliers Z is α-OPR, then the clustering instance $(V \setminus Z, d, k)$ is α-PR.

[5] In case of Asym-k-center, we consider perturbations in which d' obeys triangle inequality, but not symmetry

Notation. For integer k, let $[k] = \{1, \ldots, k\}$. Throughout, we use V to denote the input set of points, and n is the number of points. For any clustering instance (including outlier instances), $S = \{c_1, \ldots, c_k\}$ denotes an optimal set of centers, and $\mathcal{C} = \{C_1, \ldots, C_k\}$ denotes the corresponding Voronoi partition, which we call optimal clusters. Further, for a point $p \in C_i$, we often interchangebly use the terms, p is *assigned/belongs* to center c_i or cluster C_i. For a clustering with outlier instance, Z denotes the optimal set of outliers. In case of k-center, we refer to the optimal clustering cost as *optimal radius*, and denote it as R_d^*.

2.2 Some useful lemmas

Here we state some intuitive and useful lemmas regarding k-center and and Asym-k-center instances. The proofs of these lemmas are fairly simple and can be found in the full version.

Recall, in the definition of perturbation resilience, we insisted that the optimal k clustering of the perturbed instance $\mathcal{I}'$ has to be same as the optimal k clustering of the original instance. It is not hard to show, that for Asym-k-center (and also for k-center), if a $k-1$ clustering of $\mathcal{I}'$ exists whose cost is at most the optimal cost of k clustering, then the instance is not perturbation resilient. Formally,

▶ **Lemma 1.** *Consider any* Asym-k-center *instance $\mathcal{I} = (V, d, k)$. Let $S = \{c_1, \ldots, c_k\}$ be an optimal set of centers, and $\mathcal{C} = \{C_1, \ldots C_k\}$ be the corresponding optimal clustering. The optimal radius is R_d^*. Suppose there exists a set of $k-1$ centers $S' = \{c'_1, \ldots, c'_{k-1}\}$, inducing the Voronoi partition $\mathcal{C}' = \{C'_1, \ldots, C'_{k-1}\}$, with cost $\texttt{cost}_d(\mathcal{C}', S') \leq R_d^*$. Then, the optimal clustering $\mathcal{C}$ is not unique.*

One common technique we use in multiple arguments, is perturbing the input instance in a structured way. The next two lemmas are related to that.

▶ **Lemma 2.** *Consider a set of points V, and let d be an asymmetric distance function defined over V. Let G be a complete directed graph on vertices V. The edge lengths in graph G are given by the function ℓ, where for any edge (u, v), $\frac{d(u,v)}{2} \leq \ell(u, v) \leq d(u, v)$. Then the distance function d', defined as the shortest path distance in graph G using ℓ, is a metric*[6] *2-perturbation of d.*

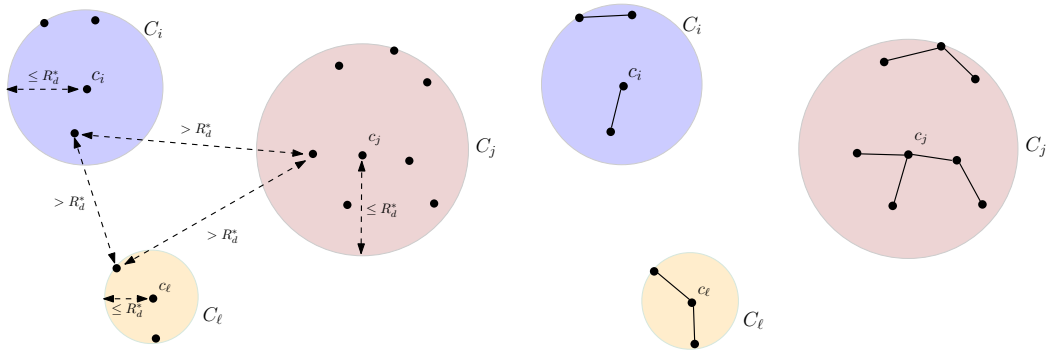
▶ **Lemma 3.** *Consider an* Asym-k-center *instance $\mathcal{I} = (V, d, k)$, and let $\mathcal{C}$ be the optimal clustering and R_d^* be the optimal radius. Let G be a complete directed graph over vertex set V. The edge lengths in graph G are given by the function ℓ, where (1) for a subset of edges E', $\ell(u, v) = \min\{d(u, v), R_d^*\}$; (2) for every other edge, $\ell(u, v) = d(u, v)$. Suppose d' is defined as the shortest path distance in graph G using ℓ. Consider the* Asym-k-center *instance $\mathcal{I}' = (V, d', k)$, let $R_{d'}^*$ be the optimal radius. If $\mathcal{C}$ is an optimal clustering in $\mathcal{I}'$, then $R_d^* = R_{d'}^*$.*

We can prove similar lemmas for the undirected version (corresponding to k-center) as well (formally stated in the full version).

3 k-center under Perturbation Resilience

In this section, we show that the natural LP relaxation for a 2-perturbation resilient k-center has an integral optimum solution.

[6] satsifies triangle inequality, and not necessarily symmetry



(a) Points in different optimal clusters are R_d^* apart. **(b)** Graph G_R for $R < R_d^*$.

Figure 1 Optimal Clusters in a 2-perturbation resilient k-center instance.

Properties of 2-perturbation resilient k-center instance: Angelidakis et al. [4] showed that in the optimal clustering of a 2-perturbation resilient k-center instance, every point is closer to its assigned center than to any point in a different cluster. In fact they show this property for general center based objectives, not just k-center. However for k-center we can show stronger structural properties : (1) any point is closer to a point in its own cluster, than to a point in a different cluster; (2) the distance between two points in two different clusters is atleast the optimal radius (see Figure 1a). We now formally state the properties (the proofs are in the full version).

▶ **Lemma 4.** *Consider a 2-perturbation resilient k-center instance $\mathcal{I} = (V, d, k)$. Let $S = \{c_1, \ldots, c_k\}$ be an optimal set of centers, and $\mathcal{C} = \{C_1, \ldots C_k\}$ be the corresponding unique optimal clustering. Consider any cluster C_i with $|C_i| \geq 2$, and let p, w be any two points in C_i. For any point q in a different cluster C_j $(i \neq j)$, we have $d(p, q) > d(p, w)$.*

▶ **Lemma 5.** *Consider a 2-perturbation resilient k-center instance $\mathcal{I} = (V, d, k)$. Let $S = \{c_1, \ldots, c_k\}$ be an optimal set of centers, and $\mathcal{C} = \{C_1, \ldots C_k\}$ be the corresponding unique optimal clustering. The optimal radius is R_d^*. Consider any point $p \in V$, and let $p \in C_i$. For any point q in a different cluster C_j $(i \neq j)$, we have $d(p, q) > R_d^*$.*

To prove the properties we use the result of Balcan et al. [12] – *any* 2-approximation algorithm for k-center finds the optimal clustering for a 2-perturbation resilient instance. They proved this result under the stronger definition of non-metric perturbation resilience, which was subsequently extended to metric perturbation resilience in an unpublished follow-up paper [14].

3.1 LP Integrality

Now, we show that as a consequence of Lemma 5, the LP relaxation for k-center is integral. Given an instance $\mathcal{I} = (V, d, k)$ of k-center and a parameter $R \geq 0$, we define the graph (also called *threshold* graph) $G_R = (V, E_R)$, where $E_R = \{(u, v) : u, v \in V, d(u, v) \leq R\}$. For a vertex v, let $\texttt{Nbr}[v] = \{u : (u, v) \in E_R\} \bigcup \{v\}$ be the neighbors (including itself). Observe, for any $R \geq R_d^*$, where R_d^* is the optimal solution cost of $\mathcal{I}$, there exists a set of k centers $S \subseteq V$, such that S *covers* V in G_R, i.e. $\bigcup_{c \in S} \texttt{Nbr}[c] = V$. Given a parameter R, we can define the following LP on graph G_R. We use y_v as an indicator variable for open centers, and x_{uv} to denote if v is assigned to u.

$$\sum_{u \in V} y_u \leq k \qquad \textbf{(kc-LP)}$$
$$x_{uv} \leq y_u \qquad \forall v \in V, u \in V$$
$$\sum_{u \in \texttt{Nbr}[v]} x_{uv} \geq 1 \qquad \forall v \in V$$
$$y_v, x_{uv} \geq 0$$

The minimum R for which **kc-LP** is feasible provides a lower bound on the optimum solution, and is the standard relaxation for k-center. It easy to see for all $R \geq R_d^*$ **kc-LP** is feasible. Further, it is well-known that the integrality gap is 2, that is, for all $R < R_d^*/2$, the LP is infeasible. However, if the k-center instance is 2-perturbation resilient, we can show that LP has no integrality gap.

▶ **Theorem 6.** *Consider a 2-perturbation resilient instance $\mathcal{I} = (V, d, k)$ of k-center. Let R_d^* be the cost of the optimal solution. Then, for any $R < R_d^*$, **kc-LP** is infeasible.*

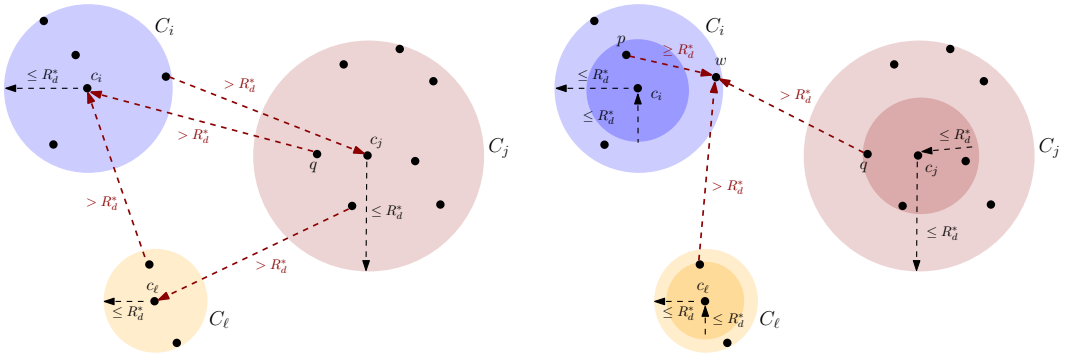
Proof. Let $C_1, \ldots C_k$ be the unique optimal clustering of instance $\mathcal{I} = (V, d, k)$, with optimal radius R_d^*. Consider an arbitrary $R < R_d^*$, and let G_R denote the corresponding threshold graph. Recall, in graph G_R the vertex set is V, and the edge set $E_R = \{(u, v) : d(u, v) \leq R\}$. According to Lemma 5, in a 2-PR instance, two points belonging to two different optimal clusters are separated by a distance of strictly more than R_d^*. Since $\mathcal{I}$ is 2-PR, graph G_R has a simple structure – for any $v \in C_i, i \in [k]$, $\texttt{Nbr}[v] \subseteq C_i$. Or in other words, the connected components of G_R are subsets of the optimal clusters (see Figure 1b).

Suppose, the k-center LP (**kc-LP**) defined over graph G_R is feasible, and (x, y) is the feasible fractional solution. Since every point is fully covered, and it can be covered only by its neighbors in G_R, we have, for all C_i, $\sum_{u \in C_i} y_u \geq 1$. Since, $\sum_{v \in V} y_v \leq k$, and the clusters $C_1, \ldots, C_k$ are disjoint, we have $\sum_{u \in C_i} y_u = 1$, for each i.

From the definition of R_d^*, there is an optimum cluster C_t such that $\min_{c \in C_t} \max_{v \in C_t} d(c, v) = R_d^*$. Let $C_t' = \{u \in C_t : y_u > 0\}$. As we argued earlier, $\sum_{u \in C_t} y_u = \sum_{u \in C_t'} y_u = 1$. Further, since for every $v \in C_t$, $\texttt{Nbr}[v] \subseteq C_t$, and v needs to be covered, we must have $C_t' \subseteq \texttt{Nbr}[v]$. Consider any $c \in C_t'$. Note that for every $v \in C_t$, c is a neighbor of v in graph G_R, i.e. $d(c, v) \leq R < R_d^*$. This implies, $\max_{v \in C_t} d(c, v) < R_d^*$ which is a contradiction. ◀

4 LP Integrality of Asym-k-center under Perturbation Resilience

We start with an LP relaxation for Asym-k-center problem by considering an unweighted directed graph on node set V. Specifically, for a parameter $R \geq 0$, we define the directed graph $G_R = (V, E_R)$, where $E_R = \{(u, v) : u, v \in V, d(u, v) \leq R\}$. For a node v, let $\texttt{Nbr}^-[v] = \{u : (u, v) \in E_R\} \bigcup \{v\}$ denote the in-neighbors, and $\texttt{Nbr}^+[v] = \{u : (v, u) \in E_R\} \bigcup \{v\}$ be the out-neighbors (including itself). Observe, for any $R \geq R_d^*$, there exists a set of k centers $S \subseteq V$, such that S *covers* V in G_R, i.e. $\bigcup_{c \in S} \texttt{Nbr}^+[c] = V$. Thus, given a parameter R, we can define the following LP relaxation on graph G_R. We use y_v as an indicator variable for open centers, and x_{uv} to denote if v is assigned to u.



(a) Cluster centers are atleast R_d^* away from points in different cluster.

(b) Cluster points R_d^* away from cluster core points, are R_d^* away from different cluster core points.

Figure 2 Properties of a 2-perturbation resilient Asym-k-center instance.

$$\sum_{u \in V} y_u \leq k \qquad \textbf{(asym-kc-LP)}$$

$$x_{uv} \leq y_u \qquad \forall v \in V, u \in V$$

$$\sum_{u \in \texttt{Nbr}^-[v]} x_{uv} \geq 1 \qquad \forall v \in V$$

$$y_v, x_{uv} \geq 0$$

For Asym-k-center, Archer [5] showed that the integrality gap is atmost $O(\log^* k)$, Infact it is tight within a constant factor [23]. The main result of the section is captured by the following theorem.

▶ **Theorem 7.** *Let $\mathcal{I} = (V, d, k)$ be a 2-perturbation resilient instance of* Asym-k-center *and let R_d^* be the cost of the optimal solution. Then, for any $R < R_d^*$, **asym-kc-LP** is infeasible.*

4.1 Properties of 2-perturbation resilient Asym-k-center instance

In Section 3 we showed that the clusters in an optimal solution to a 2-PR k-center instance have a strong separation property: $d(p, q) > R_d^*$ if p, q are in different clusters. For Asym-k-center the asymmetry in the distances does not permit a such a strong and simple separation property. However, we can show slightly weaker properties: (1) every optimal center is separated from any point in a different cluster by at least R_d^*; (2) points in a cluster which are far off from *core* points (these points have small distance "to" correponding cluster centers) in the cluster, are well-separated from core points of other clusters as well (See Figure 2a, Figure 2b). These properties suffice to prove our desired theorem. We now formally state the properties (the proofs are in the full version).

▶ **Lemma 8.** *Consider a 2-perturbation resilient* Asym-k-center *instance $\mathcal{I} = (V, d, k)$. Let $\mathcal{C} = \{C_1, \ldots C_k\}$ be the unique optimal clustering, induced by a set of centers $S = \{c_1, \ldots, c_k\}$. Let the optimal radius be R_d^*. Consider any center c_i. Then for any point q in a different cluster C_j $(i \neq j)$, we have $d(q, c_i) > R_d^*$.*

The next lemma formalizes the notion of core points and the property they enjoy.

▶ **Lemma 9.** *Consider a 2-perturbation resilient* Asym-k-center *instance $\mathcal{I} = (V, d, k)$. Let $\mathcal{C} = \{C_1, \ldots C_k\}$ be the unique optimal clustering induced by a set of centers $S = \{c_1, \ldots, c_k\}$. Let the optimal radius is R_d^*. Suppose $p \in C_i$ and $q \in C_j$ where $i \neq j$ and $d(p, c_i) \leq R_d^*$ and $d(q, c_j) \leq R_d^*$. Then for any $w \in C_i$ such that $d(p, w) \geq R_d^*$ we have $d(q, w) > R_d^*$.*

4.2 Proof of Theorem 7

Let $C_1, \ldots C_k$ be the unique optimal clustering of instance $\mathcal{I} = (V, d, k)$, with optimal radius R_d^*. Consider an arbitrary $R < R_d^*$, and let G_R denote the corresponding threshold graph. Recall, graph G_R is a directed graph defined over vertex set V, and the edge set $E_R = \{(u, v) : d(u, v) \leq R\}$. Suppose, the Asym-$k$-center LP (**asym-kc-LP**) defined over graph G_R is feasible, and (x, y) is a feasible fractional solution.

From Lemma 8, in a 2-PR instance, we have the following: if $q \notin C_i$ then $d(q, c_i) > R_d^* > R$. This implies that, in the graph G_R, for any $c_i, i \in [k]$, $\texttt{Nbr}^-[c_i] \subseteq C_i$. Let $C_i' = \{u \in \texttt{Nbr}^-[c_i] : y_u > 0\}$. Since (x, y) is a feasible solution, we must have $\sum_{u \in C_i'} y_u \geq 1$. Since, $\sum_{v \in V} y_v \leq k$, and the clusters $C_1, \ldots, C_k$ are disjoint, we have $\sum_{u \in C_i} y_u = \sum_{u \in C_i'} y_u = 1$, for all $i \in [k]$.

From the definition of R_d^* there must be a cluster C_t such that $\min_{c \in C_t} \max_{v \in C_t} d(c, v) = R_d^*$. Consider its center c_t and let $p \in C_t'$. Clearly $d(p, c_t) \leq R < R_d^*$. Furthermore, since C_t is the largest radius cluster, there exists $w \in C_t$, such that $d(p, w) \geq R_d^*$. Therefore in graph G_R, $p \notin \texttt{Nbr}^-[w]$. For any other cluster C_j, by Lemma 9, for any point $q \in C_j'$, we have $d(q, w) > R_d^*$. That is, $\texttt{Nbr}^-[w] \bigcap C_j' = \emptyset$, for any $j \neq t$. This implies w can be covered only by points that belong to C_t'. Therefore $\sum_{u \in \texttt{Nbr}^-[w]} x_{uw} \leq \sum_{u \in C_t' - p} y_u < 1$ since $y_p > 0$. This contradicts feasibility of (x, y).

5 LP Integrality of k-center-outlier under Perturbation Resilience

In this section we now consider the k-center-outlier problem. Recall that an instance $\mathcal{I} = (V, d, k, z)$ consists of a finite metric space (V, d) an integer k specifying the number of centers and an integer $z < |V|$ specifying the number of outliers that are allowed. One can write a natural LP relaxation for this problem as follows. As before, for a parameter $R \geq 0$, we define the graph $G_R = (V, E_R)$, where $E_R = \{(u, v) : u, v \in V, d(u, v) \leq R\}$. For a node v, let $\texttt{Nbr}[v] = \{u : (u, v) \in E_R\} \cup \{v\}$ be the neighbors (including itself). Observe, for any $R \geq R_d^*$, there exists a set of k centers $S \subseteq V$, and a set of outliers Z with $|Z| \leq z$, such that S *covers* $V \setminus Z$ in G_R, i.e. $\cup_{c \in S} \texttt{Nbr}[c] = V \setminus Z$. Thus, given a parameter R, we can define the following LP relaxation on graph G_R. We use y_v as an indicator variable for open centers, and x_{uv} to denote if v is assigned to u.

$$\sum_{u \in V} y_u \leq k \qquad \textbf{(kco-LP)}$$

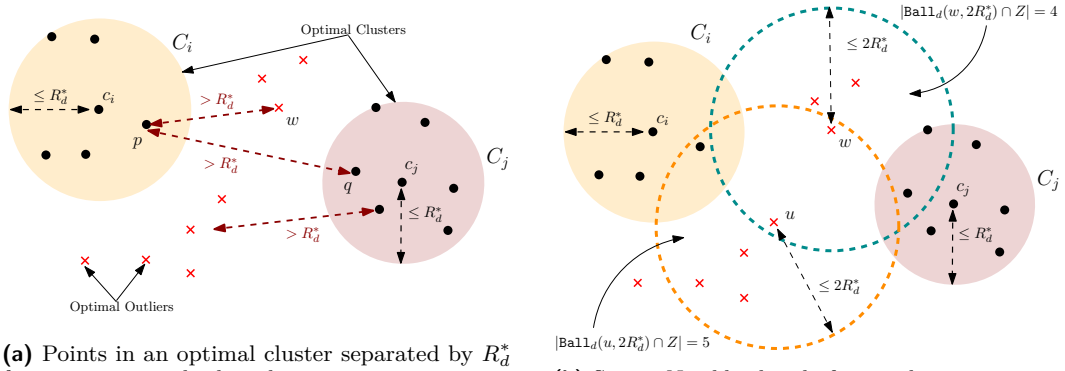
$$x_{uv} \leq y_u \qquad \forall v \in V, u \in V$$

$$\sum_{u \in V} x_{uv} \leq 1 \qquad \forall v \in V$$

$$\sum_{v \in V} \sum_{u \in V} x_{uv} \geq n - z$$

$$x_{uv} = 0 \qquad \forall v \in V, u \notin \texttt{Nbr}[v]$$

$$y_v, x_{uv} \geq 0$$



(a) Points in an optimal cluster separated by R_d^* from points outside that cluster.

(b) Sparse Neighborhood of an outlier.

Figure 3 Properties of a 2-perturbation resilient k-CENTER-OUTLIER instance.

The **kco-LP** is feasible for all $R \geq R_d^*$. The main theorem we prove in this section is as follows:

▶ **Theorem 10.** *Given a 2-perturbation resilient instance $\mathcal{I} = (V, d, k, z)$ of k-CENTER-OUTLIER with optimal cost R_d^*, **kco-LP** is infeasible for any $R < R_d^*$.*

5.1 Properties of 2-perturbation resilient k-center-outlier instance

For k-CENTER-OUTLIER we extend the properties from Section 3 that hold for 2-perturbation resilient instances. The first property shows that if p is a non-outlier point p and q is any point not in the same cluster as p (q could be an outlier) then $d(p, q) > R_d^*$. The second property is that for any outlier point q, the number of outliers in a ball of radius $2R_d^*$ is small. Specifically the number of points is strictly smaller than the size of the smallest cluster in the optimum clustering. This property makes intuitive sense, for otherwise q can define another cluster with outlier points and contradict the uniqueness of the clustering in after perturbation. We formally state them below after setting up the required notation.

Let $\mathcal{I} = (V, d, k, z)$ be a 2-outlier perturbation resilient k-CENTER-OUTLIER instance. Let $\mathcal{C} = \{C_1, \ldots, C_k\}$ be the optimum clustering, and Z be the set of outliers in the optimal solution of $\mathcal{I}$. Further, let $S = \{c_1, \ldots, c_k\}$ be the optimal centers inducing the clustering $\mathcal{C}$. Let the optimal cost be R_d^*. For each optimal cluster C_i, $n_i = |C_i|$ denotes its cardinality. Additionally, given a point $u \in Z$, and radius R, let $\texttt{Ball}_d(u, R) = \{v \in V : d(u, v) \leq R\}$ be the set of points in a ball of radius R centered at u.

The two main structural properties of an 2-OPR k-CENTER-OUTLIER instance we show are as follows (See Figure 3a, Figure 3b):

▶ **Lemma 11.** *Consider any non-outlier point $p \in V \setminus Z$, and let $p \in C_i$. For all $q \notin C_i$, $d(p, q) > R_d^*$.*

▶ **Lemma 12.** *For any outlier $p \in Z$, we have $|\texttt{Ball}_d(p, 2 \cdot R_d^*) \bigcap Z| < \min\{n_1, \ldots, n_k\}$.*

We observe that a much weaker version of the preceding lemma suffices for our proof of LP integrality. The weaker version states that $|\texttt{Ball}_d(p, R_d^*) \bigcap Z| < \min\{n_1, \ldots, n_k\}$. For if the statement is false, we could replace the smallest cluster with the cluster $\texttt{Ball}_d(p, R_d^*)$; this gives an alternate clustering with at most z outliers and the same optimum radius contradicting the uniqueness of the optimum solution.

5.2 Integrality Gap and Proof of Theorem 10

In this section, we show that **kco-LP** is infeasible for $R < R_d^*$. Recall in Lemma 11, we showed that the optimal clusters are well-separated from each other and also from the outliers. Therefore, in graph G_R, the connected components are either subsets of optimal clusters or outliers. As a consequence, in a fractional solution, non-outlier points can only be covered by points inside the cluster, and similarly outliers can be covered by outliers only. However, unlike k-center, here the tricky part is, the fractionally open outliers can potentially cover a lot of points. We show that this in fact is not possible because of the sparsity of an outlier's neighborhood.

Suppose the claim is not true, that is for some $R < R_d^*$, **kco-LP** has a feasible solution (x^*, y^*). Let $\mathcal{C} = \{C_1, \ldots, C_k\}$ be the set of clusters and Z be the outliers in the unique optimal solution of $\mathcal{I}$.

First, let us consider the simpler case when $y^*(Z) = 0$. Recall Lemma 11, for every $p \in C_i$, $(i \in [k])$, the distance to any $q \notin C_i$ is more than R_d^*. In other words, for any $p \in C_i$, $\texttt{Nbr}[p] \subseteq C_i$, and for any $w \in Z$, $\texttt{Nbr}[w] \bigcap V \setminus Z = \emptyset$. Therefore, $y^*(Z) = 0$ and the LP constraint $x_{uv} = 0, \forall v \in V, u \notin \texttt{Nbr}[v]$ implies (1) for any $w \in Z$, $x^*_{uw} = 0$ for all $u \in V$; (2) for any $v \in V \setminus Z$, and $w \in Z$, $x^*_{wv} = 0$. Therefore, (x^*, y^*) [restricted to $V \setminus Z$] is a feasible fractional solution for **kc-LP** defined for the k-center instance $\mathcal{I}' = (V \setminus Z, d, k)$, and parameter R. The optimal radius of $\mathcal{I}'$ is also R_d^*. Therefore by Theorem 6, we cannot have a feasible fractional solution for $R < R_d^*$, leading to a contradiction.

We focus on the case $y^*(Z) > 0$. Without loss of generality assume that the optimum clusters are numbered such that $n_1 \leq n_2 \leq \ldots \leq n_k$. For $i \in [k]$ let $a_i = y(C_i)$ and let $b = y^*(Z)$. For a point p let $\gamma_p = \sum_u x^*_{up}$ be the amount to which p is covered. For a set of points S we let $\gamma(S)$ denote $\sum_{p \in S} \gamma_p$.

▶ **Claim 5.1.** *Total coverage of outlier points, that is,* $\gamma(Z) = \sum_{p \in Z} \rho_p < bn_1$.

▶ **Claim 5.2.** *Let* C_i *be an optimum cluster such that* $a_i < 1$. *Then* $\gamma(C_i) \leq n_i a_i$.

Let $A = \{i \in [k] \mid a_i < 1\}$ be the indices of the clusters whose total y value is strictly less than 1. Since $y(V) = k$, we have $b \leq \sum_{i \in A}(1 - a_i)$.

Using the preceding two claims we can show that $\gamma(V) < n - z$ which contradicts the feasibility of (x^*, y^*).

6 Algorithm for k-median-outlier under Perturbation Resilience

In this section, we present a dynamic programming based algorithm for k-MEDIAN-OUTLIER, which gives an optimal solution when the instance is 2-perturbation resilient. First, we prove some structural properties of a 2-OPR k-MEDIAN-OUTLIER instance. They serve as the key ingredient in showing that our algorithm will return exact solution for 2-OPR instances.

This section is essentially a straight forward extension of the ideas in [4] once the model is set up. In a sense the model justifies the natural extension of the algorithm from [4] to the outlier setting.

6.1 Properties of 2-perturbation resilient k-median-outlier instance

Angelidakis et al. [4] proved that in the optimal clustering of a 2-perturbation resilient k-median instance, every point is closer to its assigned center than to any point in a different cluster. In the optimal solution of k-MEDIAN-OUTLIER, points are not only assigned to clusters, some points are identified as outliers as well. Here, we extend the result of [4]

to show that the optimal solution of a 2-OPR k-MEDIAN-OUTLIER instance satisfies the property: any non-outlier point is closer to its assigned center than to any point outside the cluster.

▶ **Lemma 13.** *Consider a 2-perturbation resilient k-MEDIAN-OUTLIER instance $\mathcal{I} = (V, d, k, z)$. Let $\mathcal{C} = \{C_1, \ldots C_k\}$, and Z be the unique optimal clustering and outliers resp. Consider any point $p \in V \setminus Z$, and let $p \in C_i$. For all $q \notin C_i$, we have $d(c_i, p) < d(p, q)$.*

6.2 Algorithm

In the previous section, we showed that in the optimal solution of a 2-perturbation resilient k-MEDIAN-OUTLIER instance, any non-outlier point is closer to its assigned center than to any point outside the cluster. This gives a nice structure to the optimal solution. In particular, the optimal clusters form subtrees in the minimum spanning tree over input point set. We leverage this property to design a dynamic programming based algorithm to identify the optimal clusters and outliers.

▶ **Lemma 14.** *Let $\mathcal{I} = (V, d, k, z)$ be a 2-perturbation resilient instance of the k-MEDIAN-OUTLIER problem. Let T be a minimum spanning tree on V. The optimal clusters of $\mathcal{I}$, $C_1, \ldots C_k$ are subtrees in T i.e. for any two points $p, q \in C_i$, all the points along the unique tree path between p, and q belongs to cluster C_i.*

Once we prove Lemma 14, a simple modification of the dynamic programming in [4] gives exact solution for k-MEDIAN-OUTLIER. The running time of the algorithm is $O\big((nkz)^2\big)$. Further, as noted in [4], the algorithm readily generalizes to give exact solution for other objectives like k-CENTER-OUTLIER, k-MEANS-OUTLIER, and more general ℓ_p objectives. The details can be found in the full version of the paper.

References

1. Charu C. Aggarwal. *Outlier Analysis*. Springer Publishing Company, Incorporated, 2013.
2. Sara Ahmadian, Ashkan Norouzi-Fard, Ola Svensson, and Justin Ward. Better guarantees for k-means and euclidean k-median by primal-dual algorithms. In *FOCS*, pages 61–72, 2017.
3. Sara Ahmadian and Chaitanya Swamy. Approximation Algorithms for Clustering Problems with Lower Bounds and Outliers. In *ICALP*, pages 69:1–69:15, 2016.
4. Haris Angelidakis, Konstantin Makarychev, and Yury Makarychev. Algorithms for stable and perturbation-resilient problems. In *STOC*, pages 438–451, 2017.
5. Aaron Archer. Two $O(\log^* K)$-approximation algorithms for the asymmetric k-center problem. In *IPCO*, pages 1–14, 2001.
6. David Arthur and Sergei Vassilvitskii. K-means++: The advantages of careful seeding. In *SODA*, pages 1027–1035, 2007.
7. Pranjal Awasthi, Avrim Blum, and Or Sheffet. Stability yields a ptas for k-median and k-means clustering. In *FOCS*, pages 309–318, 2010.
8. Pranjal Awasthi, Avrim Blum, and Or Sheffet. Center-based clustering under perturbation stability. *Inf. Process. Lett.*, 112(1-2):49–54, 2012. doi:10.1016/j.ipl.2011.10.006.
9. Pranjal Awasthi, Moses Charikar, Ravishankar Krishnaswamy, and Ali Kemal Sinop. The hardness of approximation of euclidean k-means. *CoRR*, abs/1502.03316, 2015. arXiv:1502.03316.
10. Pranjal Awasthi and Or Sheffet. Improved spectral-norm bounds for clustering. In *APPROX-RANDOM*, pages 37–49, 2012.

11 Maria-Florina Balcan, Avrim Blum, and Anupam Gupta. Approximate clustering without the approximation. In *SODA*, pages 1068–1077, 2009.

12 Maria-Florina Balcan, Nika Haghtalab, and Colin White. k-center clustering under perturbation resilience. In *ICALP*, pages 68:1–68:14, 2016.

13 Maria-Florina Balcan and Yingyu Liang. Clustering under perturbation resilience. In *ICALP*, pages 63–74, 2012.

14 Maria-Florina Balcan and Colin White. Clustering under local stability: Bridging the gap between worst-case and beyond worst-case analysis. *CoRR*, abs/1705.07157, 2017. arXiv:1705.07157.

15 Yonatan Bilu, Amit Daniely, Nati Linial, and Michael E. Saks. On the practically interesting instances of MAXCUT. In *STACS*, pages 526–537, 2013.

16 Yonatan Bilu and Nathan Linial. Are stable instances easy? In *ICS*, pages 332–341, 2010.

17 Johannes Blömer, Christiane Lammersen, Melanie Schmidt, and Christian Sohler. Theoretical analysis of the k-means algorithm - A survey. *CoRR*, abs/1602.08254, 2016. arXiv:1602.08254.

18 Jaroslaw Byrka, Thomas Pensyl, Bartosz Rybicki, Aravind Srinivasan, and Khoa Trinh. An improved approximation for k-median and positive correlation in budgeted optimization. *ACM Trans. Algorithms*, 13(2):23:1–23:31, 2017. doi:10.1145/2981561.

19 Deeparnab Chakrabarty, Prachi Goyal, and Ravishankar Krishnaswamy. The non-uniform k-center problem. In *ICALP*, pages 67:1–67:15, 2016.

20 Moses Charikar, Sudipto Guha, Éva Tardos, and David B. Shmoys. A constant-factor approximation algorithm for the k-median problem. In *STOC*, pages 1–10, 1999.

21 Moses Charikar, Samir Khuller, David M. Mount, and Giri Narasimhan. Algorithms for facility location problems with outliers. In *SODA*, pages 642–651, 2001.

22 Ke Chen. A constant factor approximation algorithm for k-median clustering with outliers. In *SODA*, pages 826–835, 2008.

23 Julia Chuzhoy, Sudipto Guha, Eran Halperin, Sanjeev Khanna, Guy Kortsarz, Robert Krauthgamer, and Joseph (Seffi) Naor. Asymmetric k-center is $\log^* n$-hard to approximate. *J. ACM*, 52(4):538–551, 2005. doi:10.1145/1082036.1082038.

24 Vincent Cohen-Addad and Chris Schwiegelshohn. On the local structure of stable clustering instances. In *FOCS*, pages 49–60, 2017.

25 T. F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theor. Comput. Sci.*, 38:293–306, 1985.

26 D. S. Hochbaum and D. B. Shmoys. A best possible heuristic for the k-center problem. *Mathematics of Operations Research*, 10:180–184, 1985.

27 K. Jain, M. Mahdian, E. Markakis, A. Saberi, and V. Vazirani. Greedy facility location algorithms analyzed using dual fitting with factor-revealing lp. *J. ACM*, 50(6):795–824, 2003. doi:10.1145/950620.950621.

28 Ravishankar Krishnaswamy, Shi Li, and Sai Sandeep. Constant approximation for k-median and k-means with outliers via iterative rounding. *CoRR*, abs/1711.01323, 2017. arXiv:1711.01323.

29 Amit Kumar and Ravindran Kannan. Clustering with spectral norm and the k-means algorithm. In *FOCS*, pages 299–308, 2010.

30 S. Lloyd. Least squares quantization in pcm. *IEEE Trans. Inf. Theor.*, 28(2):129–137, 2006. doi:10.1109/TIT.1982.1056489.

31 Konstantin Makarychev, Yury Makarychev, and Aravindan Vijayaraghavan. Bilu-linial stable instances of max cut and minimum multiway cut. In *SODA*, pages 890–906. SIAM, 2014.

32 Matús Mihalák, Marcel Schöngens, Rastislav Srámek, and Peter Widmayer. On the complexity of the metric TSP under stability considerations. In *SOFSEM*, pages 382–393, 2011.

33 Rafail Ostrovsky, Yuval Rabani, Leonard J. Schulman, and Chaitanya Swamy. The effectiveness of lloyd-type methods for the k-means problem. In *FOCS*, pages 165–176, 2006.

34 Rina Panigrahy and Sundar Vishwanathan. An $O(\log^* n)$ approximation algorithm for the asymmetric p-center problem. *J. Algorithms*, 27(2):259–268, 1998. doi:10.1006/jagm.1997.0921.

35 Aravindan Vijayaraghavan, Abhratanu Dutta, and Alex Wang. Clustering stable instances of euclidean k-means. In *NIPS*, pages 6503–6512, 2017.

A Related Work

There is extensive related work on clustering topics. Here we only mention some closely related work.

Clustering. For both k-center and asymmetric k-center tight approximation bounds are known. For k-center, already in the mid 1980's Gonzales [25] and Hochbaum & Shmoys [26] had developed remarkably simple 2-approximation algorithms, which are infact tight. Approximating asymmetric k-center is significantly harder. Panigrahy and Vishwanathan [34] designed an elegant $O(\log^* n)$ approximation algorithm, which was subsequently improved by Archer [5] to $O(\log^* k)$. Interestingly, the result is asymptotically tight [23].

For k-means and k-median– arguably the two most popular clustering problems – there is a long line of research (see [17] for a survey on k-means). The first constant factor approximation for the k-median problem was given by Charikar et al. [20], and the current best-known is a 2.675 approximation by Byrka et al. [18]; and it is NP-Hard to do better than $1 + 2/e \approx 1.736$ [27]. For k-means the best approximation known is 6.357 [2]. The k-means problem is widely used in practice as well, and the commonly used algorithm is Lloyd's algorithm, which is a special case of the EM algorithm [30]. While there is no explicit approximation guarantee of the algorithm, it performs remarkably well in practice with careful seeding [6] (this heuristic is called K-Means++).

Clustering with Outliers. The influential paper by Charikar et al. [21] initiated the work on clustering with outliers and other robust clustering problems. For k-center with outliers, they gave a greedy 3-approximation algorithm. Recently, it has been improved to 2-approximation [19]. A related problem of lower bounded k-center clustering with outliers has also been studied [3].

For k-median with outliers they gave a bicriteria approximation algorithm, which achieves an approximation ratio of $4(1 + \epsilon)$, violating the number of outliers by a factor of $(1 + \epsilon)$. The first constant factor approximation algorithm for this problem was given by Chen (the constant is not explicitly computed) [22]. Very recently, Krishnaswamy et al. [28] proposed a generic framework for clustering with outliers. It improves the results of Chen and gives the first constant factor approximation for k-means with outliers. However, the algorithm does not appear suitable for practice in its current form (See [1] for details on algorithms used in practice for clustering with outliers).

Perturbation Resilience. The notion of perturbation resilience was introduced by Bilu and Linial [16]. They originally considered it for the Max Cut problem, designing an exact

polynomial time algorithm for $O(n)$-stable instances [7] of Max Cut. It was later improved to $O(\sqrt{n})$-stable instances [15], and finally Makarychev et al. gave a polynomial time exact algorithm for $O(\sqrt{\log n} \cdot \log \log n)$-stable instances [31].

The definition of perturbation resilience naturally extends to clustering problems. Awasthi, Blum, and Sheffet [8] presented an exact algorithm for solving 3-perturbation resilient clustering problems with *separable center based objectives* (s.c.b.o) – this includes k-median, k-means, k-center. This result was later improved by Balcan and Liang [13], who gave an exact algorithm for clustering with s.c.b.o under $(1+\sqrt{2})$-perturbation resilience. Specifically for k-center and asymmetric k-center, Balcan, Haghtalab, and White [12] obtained a stronger result – any 2-approximation algorithm for k$-$*center* gives optimum solution for 2-perturbation resilient instances. Their result was later extended to metric perturbation resilience in a follow-up paper [14]. In fact, for k-center they gave a stronger result, that any 2-approximation algorithm for k-center can give an optimal solution for 2-perturbation resilient instances. They also showed the results are essentially tight unless NP = RP[8]. Recently, Angelidakis et al. [4], gave an unifying algorithm which gives exact solution for 2-perturbation resilient instances of clustering problems with center based objectives. In fact, their algorithms work under metric perturbation resilience, which is a weaker assumption. Perturbation resilience has also been studied in various other contexts, like TSP, Minimum Multiway Cut, Clustering with min-sum objectives [13, 31, 32].

Robust Perturbtion Resilience. Perturbation resilience requires optimal solution to remain unchanged under any valid perturbation. Balcan and Liang [13] relaxed this condition slightly, and defined (α, ϵ)-perturbation resilience (or robust perturbation resilience), in which at most ϵ fraction of the points can change their cluster membership under any α-perturbation. They gave a near optimal solution for k-median under $(2+\sqrt{3}, \epsilon)$-perturbation resilience, when the clusters are not too small. Further, for k-center and asymmetric k-center efficient algorithms are known for $(3, \epsilon)$-perturbation resilient instances, assuming mild size lower bound on optimal clusters [12].

Other Stability Notions. Several other stability models, and separation conditions have also been studied to better explain real-world instances. In a seminal paper Ostrovsky, Rabani, Schulman, and Swamy [33] considered k-means instances where the cost of clusterng using k is clusters is much lower than $k-1$ clusters. They showed, that popular K-Means++ algorithm achieves an $O(1)$-approximation for these instances. Subsequently there has been series of work many other models like approximation stability [11], distribution stability [7, 24], spectral separability [29, 10, 24], and more recently on additive perturbation stability [35].

[7] They used this name to denote perturbation resilient instances of Max Cut

[8] They showed, unless NP = RP, no polynomial-time algorithm can solve k-center under $(2-\epsilon)$-approximation stability, a notion that is stronger than perturbation resilience

Sherali-Adams Integrality Gaps Matching the Log-Density Threshold

Eden Chlamtáč[1]
Ben-Gurion University, Beer Sheva, Israel
chlamtac@cs.bgu.ac.il

Pasin Manurangsi[2]
University of California, Berkeley, USA
pasin@berkeley.edu

Abstract

The log-density method is a powerful algorithmic framework which in recent years has given rise to the best-known approximations for a variety of problems, including Densest-k-Subgraph and Small Set Bipartite Vertex Expansion. These approximations have been conjectured to be optimal based on various instantiations of a general conjecture: that it is hard to distinguish a fully random combinatorial structure from one which contains a similar planted sub-structure with the same "log-density".

We bolster this conjecture by showing that in a random hypergraph with edge probability $n^{-\alpha}$, $\tilde{\Omega}(\log n)$ rounds of Sherali-Adams cannot rule out the existence of a k-subhypergraph with edge density $k^{-\alpha-o(1)}$, for any k and α. This holds even when the bound on the objective function is lifted. This gives strong integrality gaps which exactly match the gap in the above distinguishing problems, as well as the best-known approximations, for Densest k-Subgraph, Smallest p-Edge Subgraph, their hypergraph extensions, and Small Set Bipartite Vertex Expansion (or equivalently, Minimum p-Union). Previously, such integrality gaps were known only for Densest k-Subgraph for one specific parameter setting.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis

Keywords and phrases Approximation algorithms, integrality gaps, lift-and-project, log-density, Densest k-Subgraph

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.10

Related Version A full version of the paper is available at `https://arxiv.org/abs/1804.07842`.

1 Introduction

The log-density framework [7] is an emerging technique in approximation algorithms that proposes to understand the problems of interest via an average case study. More specifically, the first step in this framework is to consider a distinguishing problem between a "random instance" and a "random instance with planted solution". Instead of considering any algorithm, the focus here is on a simple "witness counting" algorithm, in which one only counts the number of occurrences of a certain substructure (i.e. witness) of the two instances. One then uses the insights from this simple witness to devise an algorithm for worst case instances. Although at first glance this last step may seem like a large leap (i.e. from simple algorithms

[1] Partially supported by ISF grant 1002/14.
[2] Supported by NSF under Grants No. CCF 1655215 and CCF 1815434.

for average case instances to more complicated algorithms for worst case instances) and even somewhat implausible, the framework has turned out to be quite effective in tackling a number of problems, including Densest k-Subgraph (DkS) [7], Lowest Degree 2-Spanner, Smallest p-Edge Subgraph (SpES) [13], Small Set Bipartite Vertex Expansion (SSBVE) [14], Label Cover and 2-CSPs [15].

To be more concrete, let us consider the Densest k-Subgraph (DkS) problem. In DkS, we are given a graph $G = (V, E)$ and an integer k; the goal is to find subgraph of G of at most k vertices that induces a maximum number of edges. [7] considers the following distinguishing problem:

▶ **Definition 1** (Random vs Planted Subgraph Problem)**.** For certain parameters $\alpha, \beta \in (0, 1), 1 < k < n$, given a graph G, decide whether it is sampled from the Erdős-Rényi distribution $\mathcal{G}(n, n^{-\alpha})$ or from the same distribution, but with a planted subgraph on k randomly chosen vertices, sampled from $\mathcal{G}(k, k^{-\beta})$.

They noted that simple witness counting algorithms can solve the distinguishing problem w.h.p. in time $n^{O(1/\varepsilon)}$ whenever $\beta \leqslant \alpha - \varepsilon$, and fail to distinguish when $\beta \geqslant \alpha$. In particular, when $k = n^{\alpha}$, the witness counting algorithm fails to distinguish between two instances, one in which the densest k-subgraph has at most $\tilde{O}(k)$ edges, and one in which it has $\Omega(k^{1+1-\alpha}) = \Omega(k \cdot n^{\alpha(1-\alpha)})$ edges. They then describe an algorithm for general instances which achieves this exact tradeoff: when $k = n^{\alpha}$, their algorithm gives an $O(n^{\alpha(1-\alpha)+\varepsilon})$-approximation in time $n^{O(1/\varepsilon)}$.

It has been conjectured (see for example [13, 14]) that not only can the above approximation guarantee not be improved for worst-case instances, but even the Random vs Planted distinguishing problem cannot be solved below the same threshold:

▶ **Conjecture 2** (Planted Dense Subgraph Conjecture)**.** *For all $0 < \alpha < 1$ and sufficiently small $\varepsilon > 0$, and for all $k \leqslant \sqrt{n}$,*[3] *no polynomial time algorithm can solve the Random vs Planted Subgraph problem for parameters $\beta \geqslant \alpha + \varepsilon$ with non-negligible probability.*

The above conjecture demonstrates another intriguing aspect of the log-density framework: not only that the framework leads to improved algorithms, it also leads to conjectured tight hardness results. For Densest k-Subgraph, this should be contrasted with the fact that no NP-hardness of approximation is known even for a factor of say 1.01! Moreover, while inapproximability results for DkS are known under stronger complexity assumptions [19, 26, 31, 1, 9, 29], none of them achieves a ratio of the form n^{δ} for some $\delta > 0$ and hence they are still very far from giving a tight lower bound as predicted by Conjecture 2.

The importance of such tight lower bounds is also amplified by the known connections between Densest k-Subgraph and numerous other problems. In particular, there are reductions from DkS to many problems, for which either hardness of approximation is not known at all otherwise or the known hardness is very weak compared to known approximations (see e.g. [22, 23, 10, 3, 4, 17, 11, 28, 12, 20, 25]). A hardness for DkS with a concrete inapproximability ratio, such one as established by Conjecture 2, would imply strong inapproximability results for these problems as well. In addition to applications in hardness of approximation, a variant of the conjecture has also been used in public-key cryptography [2].

Despite the aforementioned applications and importance of the conjecture, little progress has been made towards actually justifying it. This is somewhat unsurprising, since, barring few

[3] Note that, while for $k \geqslant \sqrt{n}$, a spectral algorithm can distinguish the two cases beyond the log-density threshold, the best *worst-case* approximation guarantees are still at the log-density threshold even for this regime.

exceptions, it is typically hard to argue for validity of average case assumptions. A common approach to support these hypotheses is by proving lower bounds against restricted classes of algorithms, such as certain LP and SDP relaxations; this has been done, for examples, for the planted clique hypothesis [18, 30, 24, 6] and the random 3-SAT hypothesis [21, 32].

Unfortunately, even on this front, not much is known for Densest k-Subgraph. In particular, the only (non-trivial) matching lower bound shown so far is that of Bhaskara et al. [8], who showed that $\Omega(\log n/\log\log n)$ rounds of the Sherali-Adams hierarchy [33] have a matching integrality gap only for the case of $\alpha = 1/2$ and $k = \sqrt{n}$, which happens to be the parameter setting maximizing the log-density gap for DkS. No tight integrality gaps were known for other parameter settings. The situation is not better for other problems that have been studied in the log-density framework; either no lower bound of this type is known for them at all, or a lower bound is known only for one specific instantiation of parameters. Note that evidence for more specific hardness of this form is crucial in hardness-of-approximation results based on DkS and related problems, as an optimal *reduction* may use a different parameter setting than the worst-case setting for DkS (e.g. [11]), or a different problem such as SpES (e.g. [28]) or SSBVE (e.g. [17]).

1.1 Our Results

We show that for every possible parameter setting, Sherali-Adams requires a super-constant number of rounds to distinguish between a random hypergraph, and one which contains a subhypergraph with the same log-density:

▶ **Theorem 3.** *[informal; see Theorem 9] For every $c = c(n) = O(\log\log n)$, every $k \in [n]$, every constant $\varepsilon > 0$ and every $\varepsilon < \alpha < c - 1$, for the random Erdős-Rényi c-uniform hypergraph $\mathcal{G}_c(n, n^{-\alpha})$, there is a $\tilde{\Omega}(\log n)$-round Sherali-Adams solution for the existence of a subhypergraph of size at most k, and at least $k^{c-\alpha-o(1)}$ edges, where both the bound on the size and the bound on the number of edges are lifted.*

As immediate corollaries of our main theorem, we obtain integrality gaps matching the log-density threshold for a variety of problems, which we outline here. Firstly, for DkS, we have an integrality gap matching the approximation guarantee of [7] for every log-density:

▶ **Corollary 4.** *For any constant $\alpha \in (0, 1)$, for $G \sim \mathcal{G}(n, n^{-\alpha})$ and $k = n^{\alpha}$, $\tilde{\Omega}(\log n)$ rounds of Sherali-Adams applied to a Densest k-Subgraph relaxation with a lifted objective function have an integrality gap of $n^{\alpha(1-\alpha)-o(1)}$ with high probability.*

In the Smallest p-Edge Subgraph problem (SpES), we are given a graph G and a parameter p, and are asked to find a subgraph of G with p edges and a minimum number of vertices. For this problem we get integrality gaps matching the approximation guarantee of [13]:

▶ **Corollary 5.** *For any constant $\alpha \in (0, 1)$, for $G \sim \mathcal{G}(n, n^{-\alpha})$ and $p = n^{\alpha}$, $\tilde{\Omega}(\log n)$ rounds of Sherali-Adams applied to a SpES relaxation with a lifted objective function have an integrality gap of $n^{\frac{\alpha(1-\alpha)}{2-\alpha}-o(1)}$ w.h.p. In particular, for $\alpha = 2 - \sqrt{2}$, we get an integrality gap of $n^{3-2\sqrt{2}-o(1)}$.*

We remark that the algorithms in [7, 13] can be stated in terms of the Sherali-Adams relaxations used in our work. Hence, we give an essentially tight (up to a sub-polynomial factor) integrality gaps for these relaxations of DkS and SpES for every setting of parameters.

Since our main theorem works not only for random graphs but also for random hypergraphs, our integrality gaps extend to the hypergraph variants of DkS and SpES. The hypergraph

extension of DkS, called Densest k-Subhypergraph (DkSH), is to find, given a c-uniform hypergraph G and an integer k, a subhypergraph of k vertices that contains as many hyperedges as possible. Our integrality gap for Densest k-Subhypergraph is stated below.

▶ **Corollary 6.** *For any constants $c \geqslant 2$ and $\alpha \in (0, c-1)$, for $G \sim \mathcal{G}_c(n, n^{-\alpha})$ and $k = n^{\alpha/(c-1)}$, $\tilde{\Omega}(\log n)$ rounds of Sherali-Adams applied to a DkSH relaxation with a lifted objective function have an integrality gap of $n^{\alpha(1-\alpha/(c-1))-o(1)}$ w.h.p. In particular, for $\alpha = (c-1)/2$, the integrality gap is $n^{(c-1)/4-o(1)}$.*

In the Minimum p-Union (MpU) problem, the goal is to find p hyperedges whose union is as small as possible in a given c-uniform hypergraph. Our integrality gap for MpU is stated below.

▶ **Corollary 7.** *For any constants $c \geqslant 2$ and $\alpha \in (0, c-1)$, for $G \sim \mathcal{G}_c(n, n^{-\alpha})$ and $p = n^{\alpha/(c-1)}$, $\tilde{\Omega}(\log n)$ rounds of Sherali-Adams applied to a MpU relaxation with a lifted objective function have an integrality gap of $n^{\frac{\alpha(c-1-\alpha)}{(c-1)(c-\alpha)}-o(1)}$ w.h.p. In particular, for $\alpha = c - \sqrt{c}$, we get an integrality gap of $n^{1-2/(1+\sqrt{c})-o(1)}$.*

We remark that our integrality gaps for both DkSH and MpU match the log-density threshold. However, unlike their graph counterparts DkS and SpES, no approximation algorithm whose ratios match the ones predicted by the log-density framework is known for DkSH and MpU for c-uniform hypergraphs where $c \geqslant 4$. For $c = 3$, such algorithms are known only for the semi-random instances [16] but not for the general (worst case) instances.

The extension to super-constant edge size also has implications for the recently studied Small Set Bipartite Vertex Expansion (SSBVE) problem. In this problem, we are given a bipartite graph $G = (L, R, E)$ and a parameter p, and are asked to find $U \subseteq L$ of size p with a minimum number of total neighbors in R. We also match the approximation guarantee of [14] for SSBVE:

▶ **Corollary 8.** *For any constant $\alpha \in (0, 1)$, there is an infinite family of bipartite graphs, such that for any bipartite graph $G = (L, R, E)$ in this family, and $p = |L|^{\alpha}$, $\omega(1)$ rounds of Sherali-Adams applied to an SSBVE relaxation with a lifted objective function have an integrality gap of $|L|^{\alpha(1-\alpha)-o(1)}$.*

Note that [14] also gave a Sherali-Adams integrality gap for the above problem, though only for $\alpha = 1/2$ (as in the integrality gap of [8]), and without lifting the objective function.

1.2 Related Work

As mentioned earlier, Sherali-Adams gaps for Densest k-Subgraph were studied in [8]. In their paper, they focused exclusively on the case of $\mathcal{G}(n, n^{-\alpha})$ and $k = n^{\alpha}$ for $\alpha = 1/2$. They defined an LP solution which, for every small vertex set $S \subseteq V$, assigns to the variable y_S some value which depends only on the size of the smallest Steiner tree for S. As they explain, such a solution is intuitive for the specific statistical properties of $\mathcal{G}(n, n^{-1/2})$. However this intuition, and their analysis, break down for any other value of α.

We also remark that, while lower bounds for the stronger Sum-of-Square (SoS) relaxations for DkS have been studied [8, 15], the integrality gaps given do not match the gaps predicted by log-density framework; in fact, the graphs used in these results are not random graphs but rather graphs resulting from a reduction from random instances of a constraint satisfaction problem (CSP). Hence, these SoS lower bounds are unrelated to the Planted Dense Subgraph Conjecture, and it is unlikely that the approach can yield similar gaps as predicted by the conjecture.

We turn instead to more recent work on integrality gaps for lift-and-project relaxations, namely the pseudo-calibration technique. This technique was introduced by Barak et al. [6], to give tight Sum-of-Squares integrality gaps for the closely related Planted Clique problem. The pseudo-calibration heuristic involves choosing a pseudo-distribution whose low-degree Fourier coefficients match those of a random graph with a planted solution. This technique is a very powerful heuristic, however, it does not automatically guarantee that the pseudo-distribution suggested by the technique will satisfy the required constraints. It is not clear at this point whether such a solution for Densest k-Subgraph and related problems is feasible as a Sherali-Adams solution, much less Sum-of-Squares. Our solution, however, is inspired by pseudo-calibration. We are able to modify the solution suggested by pseudo-calibration in such a way that in fact it does satisfy at least the Sherali-Adams constraints. This connection is explained in Appendix A.

1.3 Our Technique

In our analysis, we must show that three kinds of lifted constraints hold: monotonicity ($y_T \leqslant y_S$ for any $S \subseteq T \subseteq V$), size constraints (bounding the number of vertices selected by the solution), and density constraints (bounding the number of edges among selected vertices).

As explained below, for every small subset $S \subseteq V$, our solution assigns to y_S the maximum value attained for some function of $G|_{S'}$ over all possible vertex sets $S' \supseteq S$ up to a certain size bound. While at a first glance, both the monotonicity constraint and the density constraint seem to follow almost immediately from the definition, it turns out that this "immediate" proof of the density constraint relies on two different size bounds for S'. To reconcile the two conditions, we show that in fact in $\mathcal{G}(n, n^{-\alpha})$, w.h.p. any optimal set S' must already be bounded in size, thus making the difference in the requirements irrelevant.

The main technical aspect of our proof involves the size constraints. As we will show, a careful analysis of the properties of optimal sets does in fact guarantee the size constraints in expectation (over the random choices in $\mathcal{G}(n, n^{-\alpha})$). However, this is not enough, since we also need to show strong concentration (especially since this must hold for all possible small subsets $S \subseteq V$). Unfortunately, the general kind of concentration result we need is not true for random graphs.[4] We again strongly rely on the properties of optimal sets, showing that they *precisely* give the condition which guarantees concentration. This is shown by bounding the left-hand-side of the LP constraint by a polynomial in the incidence vector of $\mathcal{G}(n, n^{-\alpha})$, and applying a concentration bound of Kim and Vu [27] for low-degree polynomials.

Organization. In Section 2, we will describe our solution for the Sherali-Adams relaxation and show that it satisfies the three aforementioned properties; this is the main contribution of our paper. Due to space constraint, we defer the descriptions of the relaxations for DkSH, MpU and SSBVE, and the parameters settings that give the claimed integrality gaps to the full version. Finally, we discuss several open questions in Section 3.

[4] As an example of this issue, consider the number of copies of H in $\mathcal{G}(n, n^{-3/5})$, where H is a clique of size 5 connected to a path of length 5. The expected number of copies of H in G is $n^{|V(H)|-3/5|E(H)|} = n^{10-3/5(15)} = n$. However, w.h.p. *no* copies of H will appear, since H also contains K_5, which only has an expected $n^{5-3/5(10)} = 1/n$ copies.

2 Our Sherali-Adams Solution and Its Properties

The goal of this section is to prove our main theorem, which is stated below.

▶ **Theorem 9.** *For any constants $0 < \beta, \varepsilon < 1$, any $c = c(n) = O(\log\log n)$ and any $\varepsilon < \alpha < c(n) - \varepsilon$, let $G = (V, E)$ be a hypergraph sampled from the Erdős-Rényi random c-uniform hypergraph distribution $\mathcal{G}_c(n, n^{-\alpha})$, then, w.h.p., there exists $\{y_S\}_{|S|\leqslant r}$ for some $r = \tilde{\Omega}(\log n)$ that satisfies the following:*

$$\forall S, T \subseteq V \text{ such that } |S| + |T| \leqslant r:$$

$$\sum_{i\in V}\sum_{J\subseteq T}(-1)^{|J|}y_{S\cup J\cup\{i\}} \leqslant n^{\beta+o(1)}\sum_{J\subseteq T}(-1)^{|J|}y_{S\cup J} \tag{1}$$

$$\sum_{e\in E}\sum_{J\subseteq T}(-1)^{|J|}y_{S\cup J\cup e} \geqslant n^{\beta(c-\alpha)-o(1)}\sum_{J\subseteq T}(-1)^{|J|}y_{S\cup J} \tag{2}$$

$$0 \leqslant \sum_{J\subseteq T}(-1)^{|J|}y_{S\cup J} \leqslant 1 \tag{3}$$

$$y_\emptyset = 1. \tag{4}$$

To illustrate the above constraints, it is perhaps best to think about the Densest k-Subhypergraph problem as a concrete example. In this case, y_i should be thought of as the indicator variable of whether the vertex i is selected as part of the solution (i.e. the subset of k vertices); more generally, y_S should be thought of as an indicator variable whether S is contained in the solution.

Constraint (1), which we refer to as the *size constraint* is the resulting lift of the constraint (i.e., "at most k vertices are selected") where $k = n^{\beta+o(1)}$. Now, recall that our objective is to maximize the number of induced hyperedges, i.e., to maximize $\sum_{e\in E} y_e$. Constraint (2) is the lift of the constraint $\sum_{e\in E} y_e \geqslant k^{(c-\alpha)-o(1)}$. That is of the requirement that the solution must induce at least $k^{(c-\alpha)-o(1)}$ hyperedges; we call this constraint the *density constraint*. In other words, the y_S's given in Theorem 9 behave as if they are a valid solution of the planted case of Conjecture 2 where a random graph from $\mathcal{G}_c(k, k^{-\alpha-o(1)})$ is planted into the instance. We remark also that in the typical LP relaxation, we also need to require that $y_e \leqslant y_i$ for all $i \in e$, but this is already implied by Constraint (3), which results from lifting the constraint $0 \leqslant y_S \leqslant 1$.

We will now proceed to prove Theorem 9, starting by describing our solution.

The Solution. For all $S \subseteq V$ such that $|S| \leqslant r(n)$, we define y_S as follows:

$$y_S = \frac{1}{L^{|S|}} \max_{\substack{S'\supseteq S\\ |S'|\leqslant\tau(n)}} n^{(1-\beta)(\alpha'|E(S')|-|S'|)} \tag{5}$$

where L is some sub-polynomial dampening factor and $\alpha' = \alpha - o(1)$ is a number slightly smaller than α; we will define them more precisely later.

For brevity, for every $S \subseteq V$, let $\Phi(S)$ be $\alpha'|E(S)| - |S|$, and let $\Phi_{\text{OPT}}(S) = \max_{S'\supseteq S, |S'|\leqslant\tau(n)} \Phi(S')$. Moreover, we say that a subset S' is a *candidate set* for S if $S' \supseteq S$ and $|S'| \leqslant \tau(n)$. Note that our solution can be rewritten simply as $y_S = \frac{1}{L^{|S|}} \cdot n^{(1-\beta)\Phi_{\text{OPT}}(S)}$.

For every $S \subseteq V$ of cardinality at most $r(n)$, we use $\mathcal{S}'(S)$ to denote the collection of all candidate sets S' of S that attain the optimum in our solution y_S defined in (5). In other words, a set $S' \supseteq S$ of size at most $\tau(n)$ belongs to $\mathcal{S}'(S)$ if and only if $\Phi(S') = \Phi_{\text{OPT}}(S)$.

Parameter Selection. The exact parameters that we will use are as follows:

- $\tau = \tau(n) = \frac{\log n}{c(\log\log n)^2} = \tilde{\Theta}(\log n)$.
- $r = r(n) = \frac{\tau(n)}{(\log\log n)^2} = \tilde{\Theta}(\log n)$.
- $L = 2\tau(n) = \tilde{\Theta}(\log n)$.
- $\alpha' = \alpha\left(\frac{1-10r(n)/\tau(n)}{1+3c\tau(n)/\log n}\right) = \left(1 - O(1/(\log\log n)^2)\right)\alpha = \alpha - o(1)$.

2.1 Dampening Factor and Simplified Constraints

Our use of the dampening factor L is exactly the same as that in [8]; namely, it will allow us to simplify all constraints to the case where $T = \emptyset$. To see this, first observe that our solution has the following strong monotonicity property:

▶ **Lemma 10.** *For every $S \subseteq V$ and $i \in V$ such that $|S \cup \{i\}| \leqslant r(n)$, we have $y_S \geqslant L \cdot y_{S\cup\{i\}}$*

Proof. Since $S' \in \mathcal{S}'(S\cup\{i\})$ is a candidate set for S, $y_S \geqslant L^{-|S|}n^{(1-\beta)\Phi(S')} = L \cdot y_{S\cup\{i\}}$. ◀

We can then arrive at the following lemma, which states that $\sum_{J\subseteq T}(-1)^{|J|}y_{S\cup J}$ is within a factor of two of y_S, as stated below. Since the proof is exactly identical to the proof[5] of Lemma 3.2 of [8], we do not repeat it here.

▶ **Lemma 11** ([8]). *For any $S, T \subseteq V$ such that $|S \cup T| \leqslant r$ and $S \cap T = \emptyset$, we have*

$$y_S \geqslant \sum_{J\subseteq T}(-1)^{|J|}y_{S\cup J} \geqslant y_S/2.$$

Hence, the above lemma readily implies (3). Moreover, as observed in [8], we can "replace" each side in the constraints (1) and (2) by the case where $T = \emptyset$ and lose a factor of at most two each time. (Note that when $S \cap T \neq \emptyset$, both sides are already zero.) In other words, since these constant factors can be absorbed in the $o(1)$ term, it suffices to show that the two constraints hold when $T = \emptyset$. That is, we only need to show the following for all $S \subseteq V$ such that $|S| \leqslant r$:

$$\sum_{i\in V} y_{S\cup\{i\}} \leqslant n^{\beta+o(1)}y_S \tag{6}$$

$$\sum_{e\in E} y_{S\cup e} \geqslant n^{\beta(c-\alpha)-o(1)}y_S \tag{7}$$

$$y_\emptyset = 1 \tag{8}$$

2.2 Subhypergraph Exceeding the Log-Density Threshold and the Size Cutoff

Our analysis will rely heavily on the fact that w.h.p. random hypergraphs do not contain subhypergraphs with density strictly above the log-density threshold:

▶ **Lemma 12.** *With high probability, $|E(S)| \leqslant |S| \cdot \frac{(1+3c\tau/\log n)}{\alpha}$ for all $S \subseteq V$ of size at most τ.*

[5] Bhaskara et al.'s proof [8] requires $L \geqslant r(n)$; this also holds for our choice of parameter.

The lemma follows easily by bounding the number of labeled k-vertex hypergraphs with a sufficiently large number of edges, and the expected number of copies of each in a random hypergraph. The full calculation is deferred to Appendix B.

Note that since we select $\alpha' < \frac{\alpha}{1+3c\tau(n)/\log n}$, we can immediately conclude that, with probability $1 - o(1)$, $\Phi(S) \leqslant 0$ for all $S \subseteq V$ of size at most $\tau(n)$, which imply that $y_\emptyset = 1$:

▶ **Corollary 13.** *With high probability,* $y_\emptyset = 1$.

More importantly, Lemma 12 implies that the value of the cutoff $\tau(n)$ in fact does not matter! Specifically, as shown below, all sets S' that attain the maximum value of S in fact has size at most $\tau'(n) = \tau(n)/10$. Note that in the proof of the following lemma we need a separation between α' and α, which is the main reason behind our choice of α'.

▶ **Lemma 14.** *With high probability,* $\mathcal{S}'(S) \subseteq \binom{V}{\leqslant \tau'(n)}$ *for all* $S \in \binom{V}{\leqslant r(n)}$.

Proof. Let us assume that the high probability event from Lemma 12 occurs. Suppose for the sake of contradiction that there is some $S' \in \mathcal{S}'(S)$ of size $|S'| \in [\tau(n)/10, \tau(n)]$. Then in particular, S' is at least as good a set for S as S itself. That is, we have

$$\begin{aligned} 0 \leqslant \Phi(S') - \Phi(S) &= \alpha'(|E(S')| - |E(S)|) - (|S'| - |S|) \\ &\leqslant \alpha'|E(S')| - (1 - |S|/|S'|)|S'| \\ &\leqslant \alpha'|E(S')| - (1 - 10r(n)/\tau(n))|S'|. \end{aligned}$$

But this implies that

$$|E(S')| \geqslant |S'| \cdot (1 - 10r(n)/\tau(n))/\alpha' \geqslant |S'| \cdot (1 + 3c\tau/\log n)/\alpha,$$

where the second inequality follows from our choice of α'. This contradicts our assumption that the high probability event in Lemma 12 occurs. ◀

2.3 Proof of The Density Constraint

We will next prove constraint (7). Assume that the high probability event in Lemma 14 holds. Let $S' \in \mathcal{S}'(S)$. By our assumption, $|S'| \leqslant \tau'(n)$, which is at most $\tau(n) - c$ when n is sufficiently large. Hence, for every hyperedge $e \in E$, the set $S' \cup e$ is a candidate set for $S \cup e$, meaning that if $e \not\subseteq S'$, then

$$y_{S\cup e} \geqslant L^{-|S|-c} n^{(1-\beta)\Phi(S'\cup e)} \geqslant L^{-|S|-c} n^{(1-\beta)(\Phi(S')+\alpha'-c)} = L^{-c} n^{(1-\beta)(\alpha'-c)} y_S.$$

Moreover, observe that w.h.p. the total number of hyperedges in G (and not contained in S') is at least $(1 - o(1)) \cdot n^{c-\alpha}$. As a result, summing the above inequality over all $e \in E$ gives

$$\sum_{e\in E} y_{S\cup e} \geqslant (1 - o(1)) L^{-c} n^{(c-\alpha)+(1-\beta)(\alpha'-c)} y_S = n^{\beta(c-\alpha)-o(1)} y_S.$$

Degree constraints. Algorithms in the log-density framework often rely on a bound on the minimum degree in (part of) an optimal solution. We note that when $\alpha < c - 1 - \varepsilon$ (that is, when G is dense enough not to have isolated vertices), our solution also satisfies the corresponding lifted constraints for every vertex. As before, due to the dampening factors, these constraints are equivalent to the following:

$$\sum_{e\in E: e\ni i} y_{S\cup e} \geqslant n^{\beta(c-1-\alpha)-o(1)} y_S \qquad \forall S \subseteq V \text{ such that } |S| \leqslant r, \forall i \in S \tag{9}$$

The proof follows much the same argument. As above, for $S' \in \mathcal{S}'(S)$, node $i \in S(\subseteq S')$, and hyperedge $e \not\subseteq S'$ that contains i, it follows by a similar calculation that $y_{S\cup e} \geqslant L^{-c+1}n^{\alpha'-c+1}$. Combining this bound with the observation that w.h.p. the number of hyperedges containing i is at least $(1-o(1))n^{c-1-\alpha}$, we obtain the above constraint.

2.4 Proof of The Size Constraint

We now turn our attention to the only remaining constraint: the size constraint (6). The proof of the size constraint is more complicated than the previous constraints and, before we get to the main argument, we will prove a couple more structural properties of the optimal candidate sets.

2.4.1 Structural Properties of Optimal Candidate Sets

Here, we make a number of useful observations regarding the structure of $\mathcal{S}'(S)$ and the sets it contains. First, observe that $\Phi(\cdot)$ is supermodular; this will be useful in the sequel.

▶ **Observation 15.** *For every $S_1, S_2 \subseteq V$, $\Phi(S_1 \cup S_2) + \Phi(S_1 \cap S_2) \geqslant \Phi(S_1) + \Phi(S_2)$.*

Proof. Observe that $|E(S_1\cup S_2)|+|E(S_1\cap S_2)| \geqslant |E(S_1)|+|E(S_2)|$ and $|S_1\cup S_2|+|S_1\cap S_2| = |S_1|+|S_2|$. Subtracting the two yields the above bound. ◀

From this point on, we will assume that the high probability event in Lemma 14 occurs, i.e., that all optimal candidate sets (for all $S \subseteq V$ of size at most $r(n)$) are of size at most $\tau'(n)$.

We call a collection $\mathcal{S} \subseteq \mathcal{P}(V)$ *union closed*, if, for every $S_1, S_2 \in \mathcal{S}$, $S_1 \cup S_2 \in \mathcal{S}$.

▶ **Lemma 16.** *For all $S \subseteq V$ of size at most $r(n)$, $\mathcal{S}'(S)$ is union closed.*

Proof. Consider any set $S_1, S_2 \in \mathcal{S}'(S)$. Since $S \subseteq S_1, S_2$ and $|S_1|, |S_2| \leqslant \tau'(n)$, $S_1 \cap S_2$ must also be a candidate set for S, which implies that $\Phi(S_1 \cap S_2) \leqslant \Phi_{\text{OPT}}(S) = \Phi(S_1) = \Phi(S_2)$. As a result, from Observation 15, we have $\Phi(S_1 \cup S_2) \geqslant \Phi(S_1) + \Phi(S_2) - \Phi(S_1 \cap S_2) \geqslant \Phi_{\text{OPT}}(S)$. Furthermore, $|S_1 \cup S_2| \leqslant |S_1| + |S_2| \leqslant 2\tau'(n) \leqslant \tau(n)$, meaning that $S_1 \cup S_2$ is a candidate set for S as well. Hence, $S_1 \cup S_2$ must also be in $\mathcal{S}'(S)$. ◀

Union closed collection of sets always contains a maximal set, which is simply the union of all the sets. When $\mathcal{S}'(S)$ is union closed, we denote its maximal set by $S'_{\max}(S)$. The following lemma relates the maximal optimal candidate set of $S \cup \{i\}$ to that of S:

▶ **Lemma 17.** *For all $S \subseteq V$ of size at most $r(n)$ and for all $i \in V$, $S'_{\max}(S) \subseteq S'_{\max}(S\cup\{i\})$.*

Proof. The proof is essentially the same as that of Lemma 16. Consider any $S'' \in \mathcal{S}'(S\cup\{i\})$. Since $S'_{\max}(S)$ is optimal for S and $S'_{\max}(S) \cap S''$ is also a candidate set for S, we have $\Phi(S'_{\max}(S)) \geqslant \Phi(S'_{\max}(S) \cap S'')$. Similar to the calculation in the previous lemma, the supermodularity of Φ implies that $\Phi(S'_{\max}(S)\cup S'') \geqslant \Phi(S'')$. Moreover, since $|S'_{\max}(S)\cup S''| \leqslant 2\tau'(n) \leqslant \tau(n)$, $S'_{\max}(S) \cup S''$ is a valid candidate set for S, meaning that $(S'_{\max}(S) \cup S'') \in \mathcal{S}'(S \cup \{i\})$ as desired. ◀

2.4.2 Setting up the Argument

We are now ready to bound $\sum_{i\in V} y_{S\cup\{i\}}$. For brevity, we write $S_{\max}$ to denote $S'_{\max}(S)$ and $S^i_{\max}$ to denote $S'_{\max}(S \cup \{i\})$ for every $i \in V$. Let $\mathcal{I}_{\text{iso}}$ be the set of $i \in V$ such that i is an isolated vertex in the subhypergraph $E(S^i_{\max})$. It is not hard to show the following:

▶ **Proposition 18.**
1. *For every $i \in \mathcal{I}_{iso} \setminus S_{\max}$, we have $S_{\max} \cup \{i\} \in \mathcal{S}'(S \cup \{i\})$.*
2. *For every $i \notin \mathcal{I}_{iso}$, no vertex outside of S is isolated in the subhypergraph $E(S^i_{\max})$.*

Proof.
1. Suppose for the sake of contradiction that $i \in \mathcal{I}_{\text{iso}} \setminus S_{\max}$ but $(S_{\max} \cup \{i\}) \notin \mathcal{S}'(S \cup \{i\})$. This implies

$$\Phi_{\text{OPT}}(S) = \Phi(S_{\max}) \leqslant \Phi(S_{\max} \cup \{i\}) - 1 < \Phi(S^i_{\max}) - 1 = \Phi(S^i_{\max} \setminus \{i\}) \leqslant \Phi_{\text{OPT}}(S),$$

   where the second inequality comes from $S_{\max} \cup \{i\} \notin \mathcal{S}'(S \cup \{i\})$ and the next equality comes from our assumption that i is an isolated vertex in $E(S^i_{\max})$ and the last inequality follows from the fact that $S^i_{\max} \setminus \{i\}$ is a candidate set for S. Thus, we have arrived at a contradiction.
2. Suppose for the sake of contradiction that $i \notin \mathcal{I}_{\text{iso}}$ but there exists $j \in S^i_{\max} \setminus S$ such that j is isolated in $E(S^i_{\max})$. From $i \notin \mathcal{I}_{\text{iso}}$, we have $j \neq i$; this means that $S^i_{\max} \setminus \{j\}$ is a candidate set for $S \cup \{i\}$ but this implies that

$$\Phi_{\text{OPT}}(S \cup \{i\}) = \Phi(S^i_{\max}) = \Phi(S^i_{\max} \setminus \{j\}) - 1 \leqslant \Phi_{\text{OPT}}(S \cup \{i\}) - 1,$$

   which is a contradiction. ◀

Using property 1 of Proposition 18, we can bound $\sum_{i \in V} y_{S \cup \{i\}}$ as follows.

▶ **Claim 19.** *The following bound holds:*

$$\sum_{i \in V} y_{S \cup \{i\}} \leqslant y_S \left(1 + n^{\beta} + \sum_{i \notin S_{\max} \cup \mathcal{I}_{iso}} \frac{n^{(1-\beta)\alpha' |E(S^i_{\max}) \setminus E(S_{\max})|}}{n^{(1-\beta)|S^i_{\max} \setminus S_{\max}|}} \right).$$

Proof.

$$\begin{aligned}
\sum_{i \in V} y_{S \cup \{i\}} &= \sum_{i \in S_{\max}} y_{S \cup \{i\}} + \sum_{i \notin S_{\max}} y_{S \cup \{i\}} \\
&= |S_{\max}| \cdot \frac{y_S}{L} + \sum_{i \notin S_{\max}} \frac{1}{L^{|S| \cup \{i\}}} \cdot \frac{n^{(1-\beta)\alpha' |E(S^i_{\max})|}}{n^{(1-\beta)|S^i_{\max}|}} \\
&\leqslant y_S \left(\frac{\tau(n)}{L} + \sum_{i \notin S_{\max}} \frac{n^{(1-\beta)\alpha' |E(S^i_{\max}) \setminus E(S_{\max})|}}{n^{(1-\beta)|S^i_{\max} \setminus S_{\max}|}} \right) \\
&\leqslant y_S \left(1 + \sum_{i \notin S_{\max}} \frac{n^{(1-\beta)\alpha' |E(S^i_{\max}) \setminus E(S_{\max})|}}{n^{(1-\beta)|S^i_{\max} \setminus S_{\max}|}} \right) \\
&\leqslant y_S \left(1 + |\mathcal{I}_{\text{iso}}| \cdot \frac{1}{n^{1-\beta}} + \sum_{i \notin S_{\max} \cup \mathcal{I}_{\text{iso}}} \frac{n^{(1-\beta)\alpha' |E(S^i_{\max}) \setminus E(S_{\max})|}}{n^{(1-\beta)|S^i_{\max} \setminus S_{\max}|}} \right) \quad \text{by Proposition 18} \\
&\leqslant y_S \left(1 + n^{\beta} + \sum_{i \notin S_{\max} \cup \mathcal{I}_{\text{iso}}} \frac{n^{(1-\beta)\alpha' |E(S^i_{\max}) \setminus E(S_{\max})|}}{n^{(1-\beta)|S^i_{\max} \setminus S_{\max}|}} \right)
\end{aligned}$$

◀

To bound the remaining sum in the above right hand side expression, we will need some additional notations and observations.

First, observe that, from the optimality of $S^i_{\max}$ for $S \cup \{i\}$, $\Phi(S^i_{\max})$ must be at least $\Phi(S_{\max} \cup \{i\}) \geqslant \Phi(S_{\max}) - 1$, which implies the following:

▶ **Observation 20.** *For every* $i \notin S_{\max}$, $\alpha'|E(S^i_{\max}) \setminus E(S_{\max})| - |S^i_{\max} \setminus S_{\max}| \geqslant -1$.

Next, we define an α'-strictly balanced (α'-s.b.) sets of hyperedges with respect to $S_{\max}$. Intuitively speaking, a set T of hyperedges is α'-s.b. with respect to $S_{\max}$ if we cannot select a subset $T' \subseteq S_{\max}$ such that we can add T' into $S_{\max}$ and create a new set with higher Φ than $\Phi(S_{\max})$. This is formalized below.

▶ **Definition 21.** For each $T \subseteq \binom{V}{c}$ such that $T \cap \binom{S_{\max}}{c} = \emptyset$, T is said to be *α'-strictly balanced* (α'-s.b.) with respect to $S_{\max} \subseteq V$ if for any $T' \subseteq T$, we have $|v(T') \setminus S_{\max}| > \alpha'|T'|$.

The optimality of $S_{\max}$ for S implies the following:

▶ **Observation 22.** *For every* $T \subseteq E_G$ *such that* $T \cap E(S_{\max}) = \emptyset$ *and* $|v(T) \setminus S_{\max}| \leqslant \tau(n) - \tau'(n)$, T *is α'-strictly balanced with respect to* $S_{\max}$.

For convenience, we say that for $S^* \subseteq V$, a hypergraph H with vertices S^* satisfies condition (*) if the following conditions hold:

- $S^* \supseteq S_{\max}$,
- $E(H) \setminus E(S_{\max})$ is α'-strictly balanced with respect to $S_{\max}$,
- $\alpha'|E_H(S^*) \setminus E(S_{\max})| - |S^* \setminus S_{\max}| \geqslant -1$
- Every $v \in S^* \setminus S_{\max}$ is not isolated in $E_H(S^*)$.

With the above two observations and the second property of Proposition 18, we can further bound the remaining sum in Claim 19 as follows.

$$\sum_{i \notin S_{\max} \cup \mathcal{I}_{\text{iso}}} \frac{n^{(1-\beta)\alpha'|E(S^i_{\max}) \setminus E(S_{\max})|}}{n^{(1-\beta)|S^i_{\max} \setminus S_{\max}|}} \leqslant \sum_{\substack{S^* \in \binom{V}{\leqslant \tau'(n)} \\ G|_{S^*} \text{ satisfies } (*)}} \sum_{\substack{i \notin S_{\max} \cup \mathcal{I}_{\text{iso}} \\ S^i_{\max} = S^*}} \frac{n^{(1-\beta)\alpha'|E(S^*) \setminus E(S_{\max})|}}{n^{(1-\beta)|S^* \setminus S_{\max}|}}$$

$$\leqslant \tau'(n) \cdot \sum_{\substack{S^* \in \binom{V}{\leqslant \tau'(n)} \\ G|_{S^*} \text{ satisfies } (*)}} \frac{n^{(1-\beta)\alpha'|E(S^*) \setminus E(S_{\max})|}}{n^{(1-\beta)|S^* \setminus S_{\max}|}}$$

Since $\tau'(n) = n^{o(1)}$, we will focus on bounding the final sum

$$\sum_{\substack{S^* \in \binom{V}{\leqslant \tau'(n)} \\ G|_{S^*} \text{ satisfies } (*)}} \frac{n^{(1-\beta)\alpha'|E(S^*) \setminus E(S_{\max})|}}{n^{(1-\beta)|S^* \setminus S_{\max}|}} \tag{10}$$

As we shall see, in expectation, this expression is bounded from above by $n^{\beta+o(1))}$, as required. However, this is not enough for us: we want this sum to be bounded by $n^{\beta+o(1)}$ for all choices of S's simultaneously. To do so, we will first prove a concentration for a fixed S and then use a union bound over all $n^{O(r(n))}$ choices of S's. We begin with an intuitive overview of the proof, before giving a more formal proof in Appendix C.1.

Bounding the Expectation. To bound (10) in expectation, we consider every possible hypergraph of size $\tau'(n)$ which could appear as a subhypergraph containing $S_{\max}$. Consider a hypergraph "template" $H_{X,T} = (S_{\max} \cup X, T \cup E(S_{\max}))$, where X are (at most) $\tau'(n) - |S_{\max}|$ dummy vertices, and $T \subseteq \binom{S_{\max} \cup X}{c} \setminus \binom{S_{\max}}{c}$. Then (10) can be bounded from above by

$$\sum_{\substack{X,T \text{ as above} \\ H_{X,T} \text{ satisfies } (*)}} \frac{n^{(1-\beta)\alpha'|T|} \cdot |\{S^* \supseteq S_{\max} : H_{X,T} \text{ is isomorphic to a subgraph of } G|_{S^*}\}|}{n^{(1-\beta)|X|}} \tag{11}$$

To bound the expectation (under the random choices of $\mathcal{G}_c(n, n^{-\alpha})$), it suffices to bound the expected number of copies of each $H_{X,T}$ in the above expression. This expectation is easily seen to be $n^{|X|-\alpha|T|} \leq n^{|X|-\alpha'|T|}$, allowing us to bound the expectation of (11) by

$$\sum_{\substack{X,T \text{ as above} \\ H_{X,T} \text{ satisfies } (*)}} \frac{n^{(1-\beta)\alpha'|T|}}{n^{(1-\beta)|X|}} \cdot \frac{n^{|X|}}{n^{\alpha'|T|}} = \sum_{\substack{X,T \text{ as above} \\ H_{X,T} \text{ satisfies } (*)}} n^{\beta(|X|-\alpha'|T|)}$$

$$\leqslant \sum_{\substack{X,T \text{ as above} \\ H_{X,T} \text{ satisfies } (*)}} n^{\beta}. \qquad \text{by the third condition in (*)}$$

Since the number of possible template hypergraphs $H_{X,T}$ of size $\tau'(n)$ is again subpolynomial, in expectation, we have our bound of $n^{\beta+o(1)}$.

Proving Concentration around the Expectation. As mentioned earlier, we will formally prove that concentration holds by expressing (10) as a low-degree polynomial, and applying a concentration bound of Kim and Vu [27]. However, this application is not straightforward. Not only is a large expectation not enough for our analysis, it is not true for general sets that we would even necessarily have any meaningful concentration. We must use the fact that $S_{\max}$ is an optimal set. Specifically, that we only consider α-strictly balanced subgraphs containing $S_{\max}$.

To see why this is important, let us consider a template hypergraph $H_{X,T}$ for $S_{\max}$. Recall that the expected number of copies of $H_{X,T}$ in G is $n^{|X|-\alpha|T|}$. While it may be that $|X| > \alpha|T|$, giving us a polynomially large expectation, it is still possible (from a purely graph-theoretic perspective) that for some set X' s.t. $S_{\max} \subseteq X' \subset X$, the induced subtemplate $H_{X',T'}$ (where $T' = T \cap E(H|_{X' \cup S_{\max}})$) does not satisfy this, but rather has $|X'| < \alpha|T'|$. In this case, rather than concentration, we would have a case where with high probability there is no copy of $H_{X,T}$ (since in particular w.h.p. there is no copy of $H_{X',T'}$). This does not seem bad for us, since there are even far fewer copies than expected, and we would like to get an upper bound. However, the flip side of this situation is that with $n^{-\Omega(1)}$ probability, there are actually polynomially many times *more* copies of $H_{X,T}$ than the expectation, and a polynomially small probability is not good enough, since our bound must hold simultaneously for all $n^{\Omega(r(n))}$ small sets S.

Fortunately, the α-balanced condition precisely states that for any subtemplate $H_{X',T'}$ we in fact have $|X'| \geqslant \alpha|T'|$, so this situation will not arise. Thus concentration is not ruled out, and in fact can be formally proven using the Kim-Vu bound; due to space constraint, we defer this to Appendix C.

3 Discussions and Open Questions

Having shown a general technique for proving Sherali-Adams gaps at the log-density threshold, it would now be interesting to see if our approach can be applied to more rigidly structured problems that have been considered in the log-density framework, such as Label Cover.

Furthermore, an exciting challenge that could further bolster the conjectured hardness would be to give matching lower-bounds in the Sum-of-Squares SDP hierarchy. However, note that for some parameter regimes, such an integrality gap does not hold even for a simple SDP! As shown in [7], for $1 > \alpha > 1/2$, a simple SDP relaxation for DkS applied to $G = \mathcal{G}(n, n^{-\alpha})$ can already witness the fact that G does not contain a $k = n^{\alpha}$-subgraph with more than $n^{(1+\alpha)/2} \ll k^{2-\alpha}$ edges. On the other hand, in the worst case, the best currently-known approximation for this value of k is still $k^{1-\alpha}$ (matching the log-density

threshold). Thus, at least for certain parameter regimes, it would be very interesting to prove either of the following two possibilities, both of which are consistent with the current state of our knowledge:

1. There are algorithms for certain parameter regimes which give (worst-case) approximation guarantees strictly better than the log-density gap.
2. There is some other family of instances (other than Erdős-Rényi (hyper)graphs) for which there is a hardness of approximation, or at least Sum-of-Squares integrality gap, matching the log-density threshold.

References

1 Noga Alon, Sanjeev Arora, Rajsekar Manokaran, Dana Moshkovitz, and Omri Weinstein. Inapproximabilty of densest k-subgraph from average case hardness. Unpublished Manuscript, 2011.

2 Benny Applebaum, Boaz Barak, and Avi Wigderson. Public-key cryptography from different assumptions. In *STOC*, pages 171–180, 2010. doi:10.1145/1806689.1806714.

3 Sanjeev Arora, Boaz Barak, Markus Brunnermeier, and Rong Ge. Computational complexity and information asymmetry in financial products. *Commun. ACM*, 54(5):101–107, 2011. doi:10.1145/1941487.1941511.

4 Pranjal Awasthi, Moses Charikar, Kevin A. Lai, and Andrej Risteski. Label optimal regret bounds for online local learning. In *COLT*, pages 150–166, 2015. URL: http://jmlr.org/proceedings/papers/v40/Awasthi15a.html.

5 Kazuoki Azuma. Weighted sums of certain dependent random variables. *Tohoku Mathematical Journal, Second Series*, 19(3):357–367, 1967.

6 Boaz Barak, Samuel B. Hopkins, Jonathan Kelner, P. Kothari, Ankur Moitra, and Aaron Potechin. A nearly tight sum-of-squares lower bound for the planted clique problem. In *FOCS*, pages 428–437, 2016.

7 Aditya Bhaskara, Moses Charikar, Eden Chlamtac, Uriel Feige, and Aravindan Vijayaraghavan. Detecting high log-densities: an $O(n^{1/4})$ approximation for densest k-subgraph. In *STOC*, pages 201–210, 2010. doi:10.1145/1806689.1806718.

8 Aditya Bhaskara, Moses Charikar, Aravindan Vijayaraghavan, Venkatesan Guruswami, and Yuan Zhou. Polynomial integrality gaps for strong SDP relaxations of densest k-subgraph. In *SODA*, pages 388–405, 2012. URL: http://dl.acm.org/citation.cfm?id=2095116.2095150.

9 Mark Braverman, Young Kun-Ko, Aviad Rubinstein, and Omri Weinstein. ETH hardness for densest-k-subgraph with perfect completeness. In *SODA*, pages 1326–1341, 2017.

10 Moses Charikar, MohammadTaghi Hajiaghayi, and Howard J. Karloff. Improved approximation algorithms for label cover problems. *Algorithmica*, 61(1):190–206, 2011. doi:10.1007/s00453-010-9464-3.

11 Moses Charikar, Yonatan Naamad, and Anthony Wirth. On approximating target set selection. In *APPROX*, pages 4:1–4:16, 2016. doi:10.4230/LIPIcs.APPROX-RANDOM.2016.4.

12 Stephen R. Chestnut and Rico Zenklusen. Hardness and approximation for network flow interdiction. *Networks*, 69(4):378–387, 2017. doi:10.1002/net.21739.

13 Eden Chlamtac, Michael Dinitz, and Robert Krauthgamer. Everywhere-sparse spanners via dense subgraphs. In *FOCS*, pages 758–767, 2012. doi:10.1109/FOCS.2012.61.

14 Eden Chlamtác, Michael Dinitz, and Yury Makarychev. Minimizing the union: Tight approximations for small set bipartite vertex expansion. In *SODA*, pages 881–899, 2017. doi:10.1137/1.9781611974782.56.

15 Eden Chlamtác, Pasin Manurangsi, Dana Moshkovitz, and Aravindan Vijayaraghavan. Approximation algorithms for label cover and the log-density threshold. In *SODA*, pages 900–919, 2017. doi:10.1137/1.9781611974782.57.

16 Eden Chlamtáč and Pasin Manurangsi. Tight approximations in the semirandom log-density framework via label reduction. In preparation.

17 Julia Chuzhoy, Yury Makarychev, Aravindan Vijayaraghavan, and Yuan Zhou. Approximation algorithms and hardness of the k-route cut problem. *ACM Trans. Algorithms*, 12(1):2:1–2:40, 2016. doi:10.1145/2644814.

18 Yash Deshpande and Andrea Montanari. Improved sum-of-squares lower bounds for hidden clique and hidden submatrix problems. In *COLT*, pages 523–562, 2015. URL: http://jmlr.org/proceedings/papers/v40/Deshpande15.html.

19 Uriel Feige. Relations between average case complexity and approximation complexity. In *STOC*, pages 534–543, 2002. doi:10.1145/509907.509985.

20 Uriel Feige and Yael Hitron. The ordered covering problem. *Algorithmica*, Aug 2017. doi:10.1007/s00453-017-0357-6.

21 Dima Grigoriev. Linear lower bound on degrees of positivstellensatz calculus proofs for the parity. *Theor. Comput. Sci.*, 259(1-2):613–622, 2001. doi:10.1016/S0304-3975(00)00157-2.

22 Mohammad Taghi Hajiaghayi and Kamal Jain. The prize-collecting generalized steiner tree problem via a new approach of primal-dual schema. In *SODA*, pages 631–640, 2006. URL: http://dl.acm.org/citation.cfm?id=1109557.1109626.

23 Mohammad Taghi Hajiaghayi, Kamal Jain, Lap Chi Lau, Ion I. Mandoiu, Alexander Russell, and Vijay V. Vazirani. Minimum multicolored subgraph problem in multiplex PCR primer set selection and population haplotyping. In *ICCS*, pages 758–766, 2006. doi:10.1007/11758525_102.

24 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. In *SODA*, pages 1079–1095, 2016. doi:10.1137/1.9781611974331.ch76.

25 Jonah Kallenbach, Robert Kleinberg, and Scott Duke Kominers. Orienteering for electioneering. *Operations Research Letters*, 46(2):205–210, 2018. doi:10.1016/j.orl.2017.10.013.

26 Subhash Khot. Ruling out PTAS for graph min-bisection, dense k-subgraph, and bipartite clique. *SIAM J. Comput.*, 36(4):1025–1071, 2006. doi:10.1137/S0097539705447037.

27 Jeong Han Kim and Van H. Vu. Concentration of multivariate polynomials and its applications. *Combinatorica*, 20(3):417–434, 2000. doi:10.1007/s004930070014.

28 Euiwoong Lee. Partitioning a graph into small pieces with applications to path transversal. In *SODA*, pages 1546–1558, 2017. doi:10.1137/1.9781611974782.101.

29 Pasin Manurangsi. Almost-polynomial ratio ETH-hardness of approximating densest k-subgraph. In *STOC*, pages 954–961, 2017. doi:10.1145/3055399.3055412.

30 Raghu Meka, Aaron Potechin, and Avi Wigderson. Sum-of-squares lower bounds for planted clique. In *STOC*, pages 87–96, 2015. doi:10.1145/2746539.2746600.

31 Prasad Raghavendra and David Steurer. Graph expansion and the unique games conjecture. In *STOC*, pages 755–764, 2010. doi:10.1145/1806689.1806792.

32 Grant Schoenebeck. Linear level lasserre lower bounds for certain k-csps. In *FOCS*, pages 593–602, 2008. doi:10.1109/FOCS.2008.74.

33 Hanif D. Sherali and Warren P. Adams. A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. *SIAM J. Discrete Math.*, 3(3):411–430, 1990. doi:10.1137/0403036.

A Pseudo-Calibration for Densest k-Subhypergraph

Let $n, k \in \mathbb{N}$ and $p, q \in [0,1]$ be parameters to be chosen later.

- Let $\mathcal{G}_{\text{random}} = \mathcal{G}_c(n,p)$ denote the distribution of Erdős-Rényi random c-uniform hypergraph on n vertices where each c-size set of vertices is a hyperedge w.p. p.
- Let $\mathcal{G}_{\text{planted}}$ denote the "planted" distribution where each of the n vertices is marked as in the planted solution independently with probability k/n. For each c-size set of vertices, if all of its elements are marked as in the planted solution, then they are joined by a hyperedge with probability q. Otherwise, they are joined with probability p.

We represent a graph G as $\{\frac{1-p}{\sqrt{p(1-p)}}, \frac{-p}{\sqrt{p(1-p)}}\}^{\binom{[n]}{c}}$ where $G_e = \frac{1-p}{\sqrt{p(1-p)}}$ if the hyperedge e exists and in the graph and $G_e = \frac{-p}{\sqrt{p(1-p)}}$ otherwise. As usual, for each $T \subseteq \binom{[n]}{c}$, let $\chi_T(G) = \prod_{e \in T} G_e$. Observe that

$$\mathop{\mathbb{E}}_{G \sim \mathcal{G}_{\text{random}}} [\chi_T(G)\chi_{T'}(G)] = \begin{cases} 1 & \text{if } T = T', \\ 0 & \text{otherwise.} \end{cases}$$

Finally, we use $x_u \in \{0,1\}$ to denote the indicator variable of whether $u \in [n]$ is marked as planted. As usual, we define $x_S = \prod_{u \in S} x_u$

The Pseudo-Calibration heuristic suggests the following pseudo-distribution as a candidate solution.

$$\tilde{\mathbb{E}}[x_S](G) = \sum_{\substack{T \subseteq \binom{[n]}{c} \\ |v(T) \cup S| \leqslant \tau}} \widehat{\tilde{\mathbb{E}}[x_S]}(T)\chi_T(G)$$

where

$$\begin{aligned} \widehat{\tilde{\mathbb{E}}[x_S]}(T) = \mathop{\mathbb{E}}_{(x,G) \sim \mathcal{G}_{\text{planted}}} [x_S \chi_T(G)] &= \left(\frac{k}{n}\right)^{|v(T)\cup S|} \mathop{\mathbb{E}}_{(x,G) \sim \mathcal{G}_{\text{planted}}} [\chi_T(G) \mid x_{v(T)\cup S} = 1] \\ &= \left(\frac{k}{n}\right)^{|v(T)\cup S|} \left(\frac{q-p}{\sqrt{p(1-p)}}\right)^{|T|} \\ &\approx \left(\frac{k}{n}\right)^{|v(T)\cup S|} \left(\frac{q}{\sqrt{p}}\right)^{|T|}. \end{aligned}$$

In other words, the suggested solution is

$$\tilde{\mathbb{E}}[x_S](G) = \sum_{\substack{T \subseteq \binom{[n]}{2} \\ |v(T) \cup S| \leqslant \tau}} \left(\frac{k}{n}\right)^{|v(T)\cup S|} \left(\frac{q}{\sqrt{p}}\right)^{|T|} \chi_T(G).$$

We will further approximate this by replacing $\chi_T(G)$ with $(1/\sqrt{p})^{|T|}\mathbb{1}[T \subseteq E_G]$. This gives the solution

$$\tilde{\mathbb{E}}[x_S](G) = \sum_{\substack{T \subseteq E_G \\ |v(T) \cup S| \leqslant \tau}} \left(\frac{k}{n}\right)^{|v(T)\cup S|} \left(\frac{q}{p}\right)^{|T|}.$$

Our solution (specified in (5)) is the same as the above expression except that we change the summation to maximization, and we add the dampening factor and some $o(1)$ slack in the exponent; note that the parameters there are $p = n^{\alpha}, k = n^{\beta}, q = k^{\alpha} = n^{\alpha\beta}$.

B Proof of Lemma 12

Proof. Note that this bound holds trivially for any S of size at most $c-1$ since $|E(S)|=0$. For succinctness, let $\tilde{A}=(1+3c\tau/\log n)/\alpha$. For every $k\in\{c,\dots,\tau\}$, the number of labeled k-vertex hypergraphs H with $t(k)=\lceil k\tilde{A}\rceil$ hyperedges is at most $\binom{k^c}{t(k)}\leqslant k^{c\cdot t(k)}$. Since the expected number of copies of such an H in G is $n^{|V(H)|-\alpha|E(H)|}=n^{k-\alpha t(k)}\leqslant 2^{-3c\tau k}$, we have

$$\begin{aligned}
\Pr[\exists H \text{ as above} : H \text{ is a subhypergraph of } G] &< \sum_{H \text{ as above}} \mathbb{E}[\#\text{copies of } H \text{ in G}] \\
&\leqslant \sum_{k=c}^{\tau} \sum_{\substack{H \text{ as above} \\ |V(H)|=k}} 2^{-3c\tau k} \\
&\leqslant \sum_{k=c}^{\tau} k^{c\cdot t(k)} 2^{-3c\tau k} \\
&= \sum_{k=c}^{\tau} 2^{c(t(k)\log k - 3\tau k)} \\
&\leqslant \sum_{k=c}^{\tau} 2^{c((1+k\tilde{A})\log k - 3\tau k)} \\
&\leqslant \sum_{k=c}^{\tau} 2^{ck(\tilde{A}\log k - 2\tau)} \\
&\leqslant \sum_{k=c}^{\tau} 2^{ck(\tilde{A}\log \tau - 2\tau)}.
\end{aligned}$$

Now, observe that, for sufficiently large n, we have $\tilde{A}\log\tau\leqslant\tau$. As a result, we have

$$\Pr[\exists H \text{ as above} : H \text{ is a subgraph of } G] \leqslant \sum_{k=c}^{\tau} 2^{-ck\tau} \leqslant 2^{1-c^2\tau} \leqslant 2^{-c^2\tau/2} = o(1),$$

which concludes our proof. ◀

C Proof of The Size Constraint

C.1 Concentration of Low Degree Multilinear Polynomials.

Recall that we would like to bound (10). For each $e\in\binom{V}{c}$, let X_e denote the indicator variable $\mathbb{1}[e\in E_G]$ for $G=\mathcal{G}_c(n,n^{-\alpha})$. Then (10) can easily be bounded by a multilinear polynomial in these random variables X_e's as follows:

$$\begin{aligned}
&\sum_{\substack{S^*\in\binom{V}{\leqslant\tau'(n)} \\ G|_{S^*} \text{ satisfies } (*)}} \frac{n^{(1-\beta)\alpha'|E(S^*)\setminus E(S_{\max})|}}{n^{(1-\beta)|S^*\setminus S_{\max}|}} \\
&\leqslant \sum_{S^*\in\binom{V}{\leqslant\tau'(n)}} \frac{1}{n^{(1-\beta)|S^*\setminus S_{\max}|}} \cdot \left(\sum_{\substack{T\subseteq\binom{S^*}{c}\setminus\binom{S_{\max}}{c} \\ (S^*,T) \text{ satisfies } (*)}} \mathbb{1}[T\subseteq E_G]\cdot n^{(1-\beta)\alpha'|T|} \right)
\end{aligned}$$

$$= \sum_{\substack{T \subseteq \binom{S^*}{c} \setminus \binom{S_{\max}}{c} \\ (S^*, T) \text{ satisfies } (*)}} \frac{n^{(1-\beta)\alpha'|T|}}{n^{(1-\beta)|v(T) \setminus S_{\max}|}} \prod_{e \in T} X_e \tag{12}$$

Denote the polynomial in (12) by $F((X_e)_{e \in \binom{V}{c}})$. Note that F has degree $t \leqslant \tau'(n)/\alpha'$ since $|v(T) \cup S_{\max}| \leqslant \tau'(n)$ and T being strictly balanced implies that $|T| \leqslant |v(T) \setminus S_{\max}|/\alpha' \leqslant \tau'(n)/\alpha'$.

To show that this quantity is not too large, we will resort to the following concentration bound of low degree multilinear polynomial of Kim and Vu [27], which applies for any low degree polynomials whose *expectation* of every partial derivative is small. We note that this condition is weaker than those of some other similar inequalities, such as Azuma's [5], which requires the *maximum* (of partial derivative or some other measures of effect) to be small.

▶ **Theorem 23** (Kim-Vu Concentration Bound [27]). *Let $f(x_1, \ldots, x_N)$ be any degree-t multilinear polynomial, i.e.,*

$$f(x_1, \ldots, x_N) = \sum_{T \in \mathcal{T}} w_T \prod_{i \in T} x_i$$

where $\mathcal{T}$ is a collection of subsets of $[N]$ of size at most t and $w_T \geqslant 0$ is a non-negative weight of each $T \in \mathcal{T}$.

For each $A \subseteq [N]$, let the truncated polynomial $f_A(x_1, \ldots, x_N)$ be defined as

$$f_A(x_1, \ldots, x_N) = \sum_{\substack{T \in \mathcal{T} \\ A \subseteq T}} w_T \prod_{i \in T \setminus A} x_i.$$

(In other words, for every monomial containing A, we substitute $x_i = 1$ for all $i \in A$. And other monomials are deleted entirely from the sum. Note that f_A does not depend on x_i for any $i \in A$.)

For any independent Bernoulli random variables $X_1, \ldots, X_N$, let E_A denote $\mathbb{E}[f_A(X_1, \ldots, X_N)]$ for every $A \subseteq [N]$. Moreover, let E denote $\max_{A \subseteq [N]} E_A$. Then, for any $\lambda > 1$, we have

$$\Pr[|f(X_1, \ldots, X_N) - E_\emptyset| > E(8\lambda)^t \sqrt{t!}] < 2e^2 e^{-\lambda} n^{t-1}.$$

To apply the Kim-Vu bound, we only need to show that E is also $n^{\beta + o(1)}$ as the term that is multiplied with E is negligible. The bound on E is stated below; we remark that both the α'-strictly balancedness and the constraint $\alpha'|T| - |v(T) \setminus S_{\max}| \geqslant -1$ are needed. The latter is needed even when bounding the expectation of F (i.e. $E_\emptyset$). On the other hand, roughly speaking, the former is used to ensure that the set T' that we take the partial derivative on is not too dense; otherwise, it could increase the expectation significantly. This intuition is formalized in the following proof.

▶ **Lemma 24.** *For $F((X_e)_{e \in \binom{V}{c}})$ as defined above and for any (possibility empty) subset $T' \subseteq \binom{V}{c}$, we have $E_{T'} \leqslant \left(\tau'(n)^{c\tau'(n)/\alpha'} \tau'(n)^2/\alpha'\right) k$. In particular, this also implies that $E \leqslant \left(\tau'(n)^{c\tau'(n)/\alpha'} \tau'(n)^2/\alpha'\right) k$.*

Proof. Observe that $F_{T'}((X_e)_{e \in \binom{V}{c}})$ is exactly

$$\sum_{\substack{T' \subseteq T \subseteq \binom{V}{c} \setminus \binom{S_{\max}}{c} \\ (S^*, T) \text{ satisfies } (*)}} \frac{n^{(1-\beta)\alpha'|T|}}{n^{(1-\beta)|v(T) \setminus S_{\max}|}} \prod_{e \in T \setminus T'} X_e.$$

Hence, if $|v(T') \cup S_{\max}| \geqslant \tau'(n)$, then this is simply zero. Furthermore, since the sum is only over T that is α'-strictly balanced, the expression is also zero unless $|v(T') \setminus S_{\max}| > \alpha'|T'|$. As a result, from now on, we may assume that $|v(T') \setminus S_{\max}| > \alpha'|T'|$. For brevity, let $y = |T'|$ and $z = |v(T') \setminus S_{\max}|$.

Now, observe that $\mathbb{E}[\prod_{e \in T \setminus T'} X_e]$ is simply $n^{-\alpha|T \setminus T'|}$. In other words, we have

$$
\begin{aligned}
E_{T'} &= \sum_{\substack{T' \subseteq T \subseteq \binom{V}{c} \setminus \binom{S_{\max}}{c} \\ (S^*, T) \text{ satisfies } (*)}} \frac{n^{(1-\beta)\alpha'|T|}}{n^{(1-\beta)|v(T) \setminus S_{\max}| + \alpha|T \setminus T'|}} \\
&\leqslant \sum_{\substack{a,b \in \mathbb{N}_0 \\ b+z+|S_{\max}| \leqslant \tau'(n) \\ a \leqslant \tau'(n)/\alpha' \\ \alpha'(a+y)-(b+z) \geqslant -1}} \sum_{\substack{T' \subseteq T \subseteq \binom{V}{c} \setminus \binom{S_{\max}}{c} \\ |v(T) \setminus (v(T') \cup S_{\max})| = b \\ |T| = a+y}} \frac{n^{(1-\beta)\alpha'(a+y)}}{n^{(1-\beta)(b+z)+\alpha a}} \\
&\leqslant \sum_{\substack{a,b \in \mathbb{N}_0 \\ b+z+|S_{\max}| \leqslant \tau'(n) \\ a \leqslant \tau'(n)/\alpha' \\ \alpha'(a+y)-(b+z) \geqslant -1}} n^b \cdot \binom{\tau'(n)^c}{a} \cdot \frac{n^{(1-\beta)\alpha'(a+y)}}{n^{(1-\beta)(b+z)+\alpha a}} \\
&\leqslant \tau'(n)^{c\tau'(n)/\alpha'} \cdot \sum_{\substack{a,b \in \mathbb{N}_0 \\ b+z+|S_{\max}| \leqslant \tau'(n) \\ a \leqslant \tau'(n)/\alpha' \\ \alpha'(a+y)-(b+z) \geqslant -1}} n^b \cdot \frac{n^{(1-\beta)\alpha'(a+y)}}{n^{(1-\beta)(b+z)+\alpha' a}} \\
&= \tau'(n)^{c\tau'(n)/\alpha'} \cdot \sum_{\substack{a,b \in \mathbb{N}_0 \\ b+z+|S_{\max}| \leqslant \tau'(n) \\ a \leqslant \tau'(n)/\alpha' \\ \alpha'(a+y)-(b+z) \geqslant -1}} n^{\beta((b+z)-\alpha'(a+y))} \cdot n^{\alpha' y - z} \\
&< \tau'(n)^{c\tau'(n)/\alpha'} \cdot \sum_{\substack{a,b \in \mathbb{N}_0 \\ b+z+|S_{\max}| \leqslant \tau'(n) \\ a \leqslant \tau'(n)/\alpha' \\ \alpha'(a+y)-(b+z) \geqslant -1}} n^{\beta} \cdot n^0 \\
&\leqslant \left(\tau'(n)^{c\tau'(n)/\alpha'} \tau'(n)^2/\alpha'\right) k.
\end{aligned}
$$

◀

C.2 Putting Things Together

Note that for sufficiently large n, we have $\tau'(n) \geqslant 1/\alpha'$. Applying Theorem 23 with the above bound and with $\lambda = (\log n)^{10}$ and $t = \tau'(n)/\alpha'$, we arrive at the following bound:

$$
\begin{aligned}
\Pr\left[F((X_e)_{e \in \binom{V}{2}}) > \left(2(8\log n)^{10\tau'(n)/\alpha'} (\tau'(n))^{3+(c+1)\tau'(n)/\alpha'}\right) \cdot k\right] \\
\leqslant O(e^{-\log^{10} n + (\tau'(n)/\alpha')\log n})
\end{aligned}
$$

Together with Claim 19, the above equation implies

$$
\begin{aligned}
\Pr\left[\frac{\sum_{i \in V} y_{S \cup \{i\}}}{y_S} > \left(2 + 2(8\log n)^{10\tau'(n)/\alpha'} (\tau'(n))^{3+(c+1)\tau'(n)/\alpha'}\right) \cdot k\right] \\
\leqslant O(e^{-\log^{10} n + (\tau'(n)/\alpha')\log n}).
\end{aligned}
$$

Hence, by taking union bound over all S of size at most $r(n) \leqslant \tau'(n)$ we have

$$\Pr\left[\exists S, \frac{\sum_{i\in V} y_{S\cup\{i\}}}{y_S} > \left(2 + 2(8\log n)^{10\tau'(n)/\alpha'}(\tau'(n))^{3+(c+1)\tau'(n)/\alpha'}\right)\cdot k\right] \leqslant O(e^{-\log^{10} n + (2\tau'(n)/\alpha')\log n}).$$

Finally, observe that in our parameter selection $c\tau'(n) = o(\log n/\log\log n)$; hence, the term $\left(2 + 2(8\log n)^{10\tau'(n)/\alpha'}(\tau'(n))^{3+(c+1)\tau'(n)/\alpha'}\right)$ is $n^{o(1)}$ as desired, and $O(e^{-\log^{10} n + (2\tau'(n)/\alpha')\log n}) = e^{-\log^{10} n + o(\log^2 n)} = o(1)$. In other words, have shown that the size constraint holds for all $S \subseteq V$ of size at most $r(n)$ with high probability, which concludes our proof.

Lower Bounds for Approximating Graph Parameters via Communication Complexity

Talya Eden[1]
School of Electrical Engineering, Tel Aviv University, Tel Aviv 6997801, Israel
talyaa01@gmail.com

Will Rosenbaum
School of Electrical Engineering, Tel Aviv University, Tel Aviv 6997801, Israel
will.rosenbaum@gmail.com
https://orcid.org/0000-0002-7723-9090

Abstract

In a celebrated work, Blais, Brody, and Matulef [7] developed a technique for proving property testing lower bounds via reductions from communication complexity. Their work focused on testing properties of functions, and yielded new lower bounds as well as simplified analyses of known lower bounds. Here, we take a further step in generalizing the methodology of [7] to analyze the query complexity of graph parameter estimation problems. In particular, our technique decouples the lower bound arguments from the representation of the graph, allowing it to work with any query type.

We illustrate our technique by providing new simpler proofs of previously known tight lower bounds for the query complexity of several graph problems: estimating the number of edges in a graph, sampling edges from an almost-uniform distribution, estimating the number of triangles (and more generally, r-cliques) in a graph, and estimating the moments of the degree distribution of a graph. We also prove new lower bounds for estimating the edge connectivity of a graph and estimating the number of instances of any fixed subgraph in a graph. We show that the lower bounds for estimating the number of triangles and edge connectivity also hold in a strictly stronger computational model that allows access to uniformly random edge samples.

2012 ACM Subject Classification Theory of computation → Lower bounds and information complexity

Keywords and phrases sublinear graph parameter estimation, lower bounds, communication complexity

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.11

Related Version A full version of this paper is available on arXiv [12], `https://arxiv.org/abs/1709.04262`.

Acknowledgements We thank Dana Ron and Oded Goldreich for their discussion and commentary on previous versions of this work. We also thank the anonymous reviewers for their valuable comments.

[1] This research was partially supported by a grant from the Blavatnik fund. The author is grateful to the Azrieli Foundation for the award of an Azrieli Fellowship, and for the Weinstein Institute for their support.

1 Introduction

Since the seminal work of Yao [34], (two party) communication complexity has become a central topic in computational complexity. While communication complexity is well-studied in its own right, its importance in complexity theory is highlighted by its numerous applications, for example, in proving lower bounds for Turing machines [29, 23], streaming algorithms [4, 6], circuit complexity [24], distributed algorithms [21, 31], learning theory [32], and algorithmic game theory [8, 19]. Lower bounds via reductions from communication complexity also tend to be conceptually simple: delicate analysis is delegated to a small handful of fundamental results in communication complexity.

In typical applications of communication complexity, instances of the problem considered can be readily partitioned into two or more pieces. This not the case in the context of property testing [16], where the goal is to distinguish instances of a problem that satisfy some property from those that are "far" from satisfying the property. Thus, the work of Blais, Brody, and Matulef [7] was surprising, as it drew a close connection between communication complexity and property testing. Specifically, in [7], the authors develop a framework for applying communication complexity to obtain lower bounds in the property testing model. Their methodology yielded new results in property testing, as well as simpler proofs of known results.

The framework of [7] (which was subsequently generalized by Goldreich in [15]) applied to the property testing of functions. In this context, an algorithm is given query access to a function, where each query consists of evaluating the function at a single input specified by the algorithm. The goal is to design algorithms that distinguish functions having some property – for example being monotonic, or k-linear – from those that are far from having the property in the sense that a constant fraction of the function's outputs must be changed in order for it to satisfy the property. The basic methodology of [7] and [15] is as follows. Let P be a property, Π a two-party communication problem, and x and y the private inputs of the two parties. The idea is then to construct a function $f_{x,y}$ with the following properties:

1. if $\Pi(x, y) = 1$, then $f_{x,y}$ satisfies P;
2. if $\Pi(x, y) = 0$, then $f_{x,y}$ is far from satisfying P;
3. for each z in the domain of $f_{x,y}$, $f_{x,y}(z)$ can be computed from x and y using at most B bits of communication.

The main result of [7] (cf. [15]) states that under the three conditions above, any algorithm that tests P requires at least $R(\Pi)/B$ queries, where $R(\Pi)$ is the randomized communication complexity of Π.

In this work, our goal is to prove lower bounds on the number of queries necessary to (approximately) solve various graph problems. In particular, we adapt the framework described above to the context of graph parameter estimation problems. Any graph $G = (V, E)$ on n nodes can be viewed as a Boolean function whose values are are the entries of the adjacency matrix of G. Thus we can directly apply the methodology of [7]. This view of graph property testing corresponds to the "dense graph" model introduced in [16]: the graph is accessed only through "pair queries" (i.e., asking if two nodes share an edge), and two graphs are far apart only if they differ on εn^2 edges. However, this model is not suitable for analyzing graphs with $o(n^2)$ edges. In order to deal with property testing and parameter estimation in non-dense families of graphs, more refined graph models were introduced. These models allow additional types of queries – degree queries and neighbor queries – that cannot be handled in any obvious way using the models of [7] and [15].

1.1 Our Results

Our main result (Theorem 3.3) gives a general reduction from communication complexity problems to graph problems. The theorem is closely related to Theorem 3.1 in [15], but it makes a further step at generalizing the results of [7]. In particular, our result makes no assumptions about the representation of the objects in question or the types of queries allowed (although nontrivial lower bounds are only obtained when the allowable queries can be efficiently simulated with a 2-party communication protocol). Thus, we believe our main result offers several advantages. Since our approach decouples the graph queries from the representation of the graph (e.g., by its adjacency lists or adjacency matrix), it can handle many different query models (possibly simultaneously). Further, our framework may be useful in distinguishing the relative power of different query access models. Finally, we believe our results unify and simplify previous lower bound arguments for graph parameter estimation (which typically relied upon careful analysis of statistical distances between families of graphs).

We apply our lower bound framework to the following graph problems

1. estimating the number of edges [14, 18] (Section 4.1),
2. sampling edges from an almost-uniform distribution [13] (Section 4.2),
3. estimating the number of triangles [9] (Section 4.3) and r-cliques [10] (see the full version [12])
4. estimating the moments of the degree distribution [11] (see the full version [12])
5. estimating the number of instances of any fixed subgraph H (Section 4.1),
6. estimating the edge-connectivity of a graph (Section 4.4).

The lower bounds for 1–4 match previously known results, while we believe 5 and 6 are new. In results 3 and 6, our lower bounds hold in a strictly stronger graph access model that additionally allows uniformly random edge samples. Interestingly, all of the lower bounds we prove are polynomial in the size of the instance, whereas the lower bounds presented in [7] and [15] are typically logarithmic in the instance size.

1.2 Related Work

A model for query-based sublinear graph algorithms was first presented in the seminal work of Goldreich, Goldwasser, and Ron [16] in the context of property testing. Their model is appropriate for dense graphs, as only "pair queries" (i.e., queries of the form *"Do vertices u and v share an edge?"*) are allowed. An analogous model for bounded degree graphs was introduced by Goldreich and Ron in [17]. In this model, it is assumed that all vertices have degree at most Δ, and the graph is accessed via neighbor queries (*"Who is v's i^{th} neighbor?"* for $i \leq \Delta$). A similar model for sparse graphs was introduced by Parnas and Ron in [28], which does not assume that the maximum degree in the graph is bounded, and additionally allows degree queries (*"What is v's degree?"*). Kaufman, Krivelevich and Ron [25] introduced the general graph model which allows all of the above queries – pair, degree and neighbor queries. The lower bounds we prove all apply to the general graph model.

The problem of estimating the average degree of a graph (or equivalently, the number of edges in a graph) was first studied by Feige [14]. In [14], Feige proves tight bounds on the number of degree queries necessary to estimate the average degree. In [18], Goldreich and Ron study the same problem, but in a model that additionally allows neighbor queries. In this model, they prove matching upper and lower bounds for the number of queries needed to estimate the average degree. In Corollary 4.2 we achieve the same lower bound as [18] for estimating the number of edges in a graph (their lower bound as well as ours also holds when

allowing for pair queries). The related problem of sampling edges from an almost-uniform distribution was recently studied by Eden and Rosenbaum in [13]. They prove tight bounds on the number of queries necessary to sample an edge in a graph from an almost-uniform distribution. In Theorem 4.6 we present a new derivation of the lower bound presented in [13].

Eden et al. [9] prove tight bounds on the number of queries needed to estimate the number of triangles. Their results were generalized by Eden et al. in [10] to approximating the number of r-cliques for any $r \geq 3$. In Corollary 4.3 and Theorem 4.7, we present a new derivation of the lower bound of [9] for estimating the number of triangles. In the full version [12], we generalize the triangle lower bound construction to obtain a lower bound for r-cliques matching that of [10]. The recent work of Bera and Chakrabarti [6] proves communication complexity lower bounds for counting subgraphs in a model where the graph's edges are partitioned between two players. The authors apply this result to obtain *space* lower bounds in the streaming model, but the result also implies the query lower bound we achieve in Section 4.3 and uses similar techniques to our proof.[2]

In [20], Gonen et al. study the problem of approximating the number of s-star subgraphs, and give tight bounds on the number of (degree, neighbor, pair) queries needed to solve this problem. As noted by Eden et al. [11], counting s-stars is closely related to computing the s^{th} moment of the degree sequence. In [11], the authors provide a simpler optimal algorithm for computing the s^{th} moment of the degree sequence that has better or matching query complexity when the algorithm is also given an upper bound on the arboricity of the graph. In the full version of this paper [12], we prove the lower bounds of [11], which build upon and generalize the lower bounds of [20].

The recent paper of Aliakbarpour et al. [2] proposes an algorithm for estimating the number of s-star subgraphs that is allowed uniformly random edge samples as a basic query as well as degree queries. Interestingly, the additional computational power afforded by random edge queries allows their algorithm to break the lower bound of [20]. We remark that our lower bounds for estimating the number of triangles (Theorem 4.7) and edge connectivity (Theorem 4.9) still hold in this stronger query access model.

2 Preliminaries

2.1 Graph Query Models

Let $G = (V, E)$ be a graph where $n = |V|$ is the number of vertices and $m = |E|$ is the number of edges. We assume that the vertices V are given distinct labels, say, from $[n] = \{1, 2, \ldots, n\}$. For $v \in V$, let $\Gamma(v)$ denote the set of neighbors of v, and $\deg(v) = |\Gamma(v)|$ is v's degree. For each $v \in V$, we assume that $\Gamma(v)$ is ordered by specifying some arbitrary bijection $\Gamma(v) \to [\deg(v)]$ so that we may refer unambiguously to v's i^{th} neighbor. We let $\mathcal{G}_n$ denote the set of all graphs on n vertices, together will all possible labelings of the vertices (from $[n]$) and all orderings of the neighbors of each vertex, and we define $\mathcal{G} = \bigcup_{n \in \mathbf{N}} \mathcal{G}_n$.

We consider algorithms that access G via queries. In general, a **query** is an arbitrary function $q : \mathcal{G} \to \{0,1\}^*$. We are interested in the following question: *"Given a set Q of allowable queries and a graph problem g (e.g. a computing function, estimating a graph parameter, etc.), how many queries $q \in Q$ are necessary to compute g?"*

[2] We thank an anonymous referee for bringing [6] to our attention.

Since we associate the vertex set V with the set $[n]$, we allow algorithms to have free access to the vertex set of the graph[3] – algorithms are only charged for obtaining information about the edges of a graph. We focus on models that allow the following types of queries:

1. **degree query** $\mathrm{d} : V \to [n-1]$, where $\mathrm{d}(v)$ returns v's degree.
2. **neighbor query** $\mathrm{nbr}_i : V \to V \cup \{\varnothing\}$ for $i \in [n-1]$, where $\mathrm{nbr}_i(v)$ returns v's i^{th} neighbor if $i \leq \deg(v)$ and $\varnothing$ otherwise.
3. **pair query** $\mathrm{pair} : V \times V \to \{0,1\}$, where $\mathrm{pair}(u,v)$ returns 1 if $(u,v) \in E$ and 0 otherwise $(u,v) \notin E$.

Taking Q to be the set of all neighbor, degree, and pair queries, we have $|Q| = O(n^2)$. This query model is known as the **general graph model** introduced in [25].[4] In Theorems 4.7 and 4.9, we also consider an expanded model that allows random edges to be sampled from a uniform distribution (cf. [2]).

We wish to characterize the **query complexity** of graph problems, that is, the minimum number of queries necessary to solve the problem. We consider randomized algorithms, and we assume the randomness is provided via a random string $\rho \in \{0,1\}^{\mathbf{N}}$. Since the query complexity of the various estimation problems we consider depends on the measure being estimated, we use the expected query complexity, rather than the worst case query complexity.[5]

Most of our results are lower bounds on the number of expected queries necessary to estimate graph parameters.

▶ **Definition 2.1.** A **graph parameter** is a function $g : \mathcal{G} \to \mathbf{R}$ that is invariant under any permutation of the vertices of each $G \in \mathcal{G}$. Formally, g is a graph parameter if for every $n \in \mathbf{N}$, $G = (V,E) \in \mathcal{G}_n$ and every permutation $\pi : [n] \to [n]$, the graph $G_\pi = (V, E_\pi)$ defined by $(v_{\pi(i)}, v_{\pi(j)}) \in E_\pi \iff (v_i, v_j) \in E$ satisfies $g(G_\pi) = g(G)$.

▶ **Definition 2.2.** Let $g : \mathcal{G}_n \to \mathbf{R}$ be a graph parameter, $\mathcal{A}$ an algorithm, and $\varepsilon > 0$. We say that say that $\mathcal{A}$ **computes a (multiplicative) $(1 \pm \varepsilon)$-approximation** of g if for all $G \in \mathcal{G}$, the output of $\mathcal{A}$ satisfies $\Pr_r(|\mathcal{A}(G) - g(G)| \leq \varepsilon g(G)) \geq 2/3$. Here, the probability is taken over the random choices of the algorithm $\mathcal{A}$ (i.e., over the random string ρ).

▶ Remark 2.3. In the general graph model, every graph G can be explored using $O(\max\{n,m\})$ queries, for example, by using depth first search. Thus, we are interested in algorithms that perform $o(\max\{n,m\})$ – or even better, $o(n)$ – queries.

2.2 Communication Complexity Background

In this section, we briefly review some background on two party communication complexity and state a fundamental lower bound for the disjointness function. We refer the reader to [26] for a detailed introduction.

[3] In the sparse and general graph models of property testing, it is often assumed that the identities of the vertices are not known in advance. Rather, algorithms may sample vertices from a uniform distribution or discover new vertices that are neighbors of known vertices. Since the current paper aims to prove lower bounds, the assumption that the identities of vertices are known in advance is without loss of generality.

[4] Allowing only neighbor queries while assuming the graph has maximal degree d, and considering the distance with respect to $n \cdot d$ is known as the "bounded degree graph" model, while only allowing pair queries and considering the distance with respect to n^2 is the "dense graph" model.

[5] In communication complexity, it is customary to use worst-case complexity when analyzing randomized protocols. This is done without loss of generality, as protocol with a given expected communication cost can be converted to a protocol with asymptotically equal worst-case communication cost and slightly higher error probability. Such a transformation from expected to worst-case lower bounds is not generally possible for query complexity when the expected cost of a protocol depends on the parameter being estimated.

We consider two party communication complexity in the following setting. Let $f : \{0,1\}^N \times \{0,1\}^N \to \{0,1\}$ be a Boolean function. Suppose two parties, traditionally referred to as Alice and Bob, hold x and y, respectively, in $\{0,1\}^N$. The *(randomized) communication complexity*[6] of f is the minimum number of bits that Alice and Bob must exchange in order for both of them to learn the value $f(x,y)$.

More formally, let Π be a communication protocol between Alice and Bob. We assume that Π is randomized, and that Alice and Bob have access to a shared random string, ρ. We say that Π **computes** f if for all $x,y \in \{0,1\}^N$, $\Pr_\rho[\Pi(x,y) = f(x,y)] \geq 2/3$, where the probability is taken over all random strings ρ. For fixed inputs $x,y \in \{0,1\}^N$ and random string ρ, we denote the number of bits exchanged by Alice and Bob using Π on input (x,y) and randomness ρ by $|\Pi(x,y;\rho)|$. The **(expected) communication cost**[7] of Π is defined by $\text{cost}(\Pi) = \sup_{x,y} \mathbf{E}_\rho(|\Pi(x,y;\rho)|)$. Finally, the **(randomized) communication complexity** of f, denoted $R(f)$, is the minimum cost of any protocol that computes f: $R(f) = \min\{\text{cost}(\Pi) \mid \Pi \text{ computes } f\}$.

The notion of communication complexity extends to partial functions in a natural way. That is, we may restrict attention to particular inputs for f and allow Π to have arbitrary output for all other values. Formally, we model this extension to partial functions via **promises** on the input of f. Let $P \subseteq \{0,1\}^N \times \{0,1\}^N$. We say that a protocol Π computes f for the promise P if for all $(x,y) \in P$, $\Pr_\rho(\Pi(x,y) = f(x,y)) \geq 2/3$. The communication complexity of a promise problem (or equivalently, a partial function) is defined analogously to the paragraph above.

One of the fundamental results in communication complexity is a linear lower bound for the communication complexity of the disjointness function. Suppose Alice and Bob hold subsets $A, B \subseteq [N]$, respectively. The disjointness function takes on the value 1 if $A \cap B = \varnothing$, and 0 otherwise. By associating A and B with their characteristic vectors in $\{0,1\}^N$ (i.e. $x_i = 1 \iff i \in A$ and $y_j = 1 \iff j \in B$), we can define the disjointness function as follows.

▶ **Definition 2.4.** For any $x,y \in \{0,1\}^N$, the **disjointness function** is defined by the formula $\text{disj}(x,y) = \neg \bigvee_{i=1}^N x_i \wedge y_i$.

The following lower bound for the communication complexity of disj was initially proved by Kalyansundaram and Schintger [22] with subsequent simplified proofs by Razborov [30] and Bar-Yossef et al. [5]. All of the results we present rely upon this fundamental lower bound.

▶ **Theorem 2.5** ([22, 30])**.** *The randomized communication complexity of the disjointness function is* $R(\text{disj}) = \Omega(N)$*. This result holds even if* x *and* y *are promised to satisfy* $\sum_{i=1}^N x_i y_i \in \{0,1\}$ *– that is, Alice's and Bob's inputs are either disjoint or intersect on a single point.*

The promise in Theorem 2.5 is known as **unique intersection**. We will also use a variant of the unique intersection problem that we refer to as the *k-intersection problem.*

[6] Throughout this paper, all algorithms and protocols are assumed to be randomized.

[7] It is more common in the literature to define the cost in terms of the worst case random string ρ rather than expected. However, for our purposes (since we consider the expected query complexity) it will be more convenient to use expected cost. We allow our protocols to err with (small) constant probability, so this difference only affects the communication complexity by a constant factor.

▶ **Definition 2.6.** Let $x, y \in \{0,1\}^N$. We say that x and y are ***k*-intersecting** if $\sum_{i=1}^{N} x_i y_i \geq k$. The k-intersection function is defined by the formula $\text{int}_k(x,y) = 1$ if $\sum_i x_i y_i \geq k$ and 0 otherwise.

The following consequence of Theorem 2.5 is proven in Appendix A.

▶ **Corollary 2.7.** *$R(\text{int}_k) = \Omega(N/k)$. The result holds even if x and y are promised to satisfy $\sum_i x_i y_i \in \{0, k\}$.*

3 General Lower Bounds

In this section, we describe a framework for obtaining general query lower bounds from communication complexity. Let $\mathcal{G}_n$ denote the family of graphs on the vertex set $V = [n]$, which we assume have labels 1 through n. We will use $g : \mathcal{G}_n \to \{0,1\}$ to denote a Boolean function on $\mathcal{G}_n$.

▶ **Definition 3.1.** Let $P \subseteq \{0,1\}^N \times \{0,1\}^N$. Suppose $f : P \to \{0,1\}$ is an arbitrary (partial) function, and let g be a Boolean function on $\mathcal{G}_n$. Let $\mathcal{E} : \{0,1\}^N \times \{0,1\}^N \to \mathcal{G}_n$. We call the pair $(\mathcal{E}, g)$ an **embedding** of f if for all $(x,y) \in P$ we have $f(x,y) = g(\mathcal{E}(x,y))$.

For a general embedding $(\mathcal{E}, g)$ of a function f, the edges of $\mathcal{E}(x,y)$ can depend on x and y in an arbitrary way. In order for the embedding to yield meaningful lower bounds, however, each allowable query q should be computable from x and y with little communication.

▶ **Definition 3.2.** Let $q : \mathcal{G}_n \to \{0,1\}^*$ be a query and $(\mathcal{E}, g)$ an embedding of f. We say that q has **communication cost** at most B and write $\text{cost}_{\mathcal{E}}(q) \leq B$ if there exists a (zero-error) communication protocol Π_q such that for all $(x,y) \in P$ we have $\Pi_q(x,y) = q(\mathcal{E}(x,y))$ and $|\Pi_q(x,y)| \leq B$.

▶ **Theorem 3.3.** *Let Q be a set of allowable queries, $f : P \to \{0,1\}$, and $(\mathcal{E}, g)$ an embedding of f. Suppose that each query $q \in Q$ has communication cost $\text{cost}_{\mathcal{E}}(q) \leq B$. Suppose $\mathcal{A}$ is an algorithm that computes g using T queries (in expectation) from Q. Then the expected query complexity of $\mathcal{A}$ is $T = \Omega(R(f)/B)$.*

Proof. Suppose $\mathcal{A}$ computes g using T queries in expectation. From $\mathcal{A}$ we define a two party communication protocol Π_f for f as follows. Let x and y denote Alice and Bob's inputs, respectively, and ρ their shared public randomness. Alice and Bob both invoke $\mathcal{A}$, letting their shared randomness ρ be the randomness of $\mathcal{A}$. Whenever $\mathcal{A}$ performs a query q that Alice or Bob cannot answer on their own, they communicate to the other party in order to determine the outcome of the query.[8] That is, they invoke Π_q in order to compute the response a to query q. The protocol terminates when $\mathcal{A}$ halts and returns an answer $\mathcal{A}(G)$, at which point Alice and Bob determine their answer to f according to $\mathcal{A}(G)$.

Since $\Pr_\rho(\mathcal{A}(G) = g(G)) \geq 2/3$, and $g(G) = f(x,y)$ it is clear that Π_f computes f. Further, since each Π_q satisfies $|\Pi_q| \leq B(q)$, we have $\text{cost}(\Pi_f) = 2B \cdot T$. Since $\text{cost}(\Pi_f) \geq R(f)$, we have $T \geq R(f)/2B$, as desired. ◀

Given the above Theorem, we suggest the following framework for proving graph query lower bounds.

[8] Since the randomness of $\mathcal{A}$ is the shared randomness of Alice and Bob they both witness the same execution of $\mathcal{A}$ and agree *without communication* on which query is being performed during each step. Further, since they both know the function $\mathcal{E}$, they can individually determine if a query q cannot be answered by the other party. In this case, they invoke Π_q.

1. Choose a "hard" communication problem $f : P \to \{0,1\}$.
2. Define functions $\mathcal{E} : P \to \mathcal{G}_n$ and $g : \mathcal{G}_n \to \{0,1\}$ such that $(\mathcal{E}, g)$ is an embedding of f in the sense of Definition 3.1.
3. For each allowable query $q \in Q$, bound B, the number of bits that must be exchanged in order to simulate q given $\mathcal{E}$.

4 Lower Bounds for Particular Problems

In this section, we derive lower bounds for particular problems. In all cases, we allow Q to be the family of all degree, neighbor, and pair queries. In the case of the previously known lower bounds (estimating the number of edges, cliques, s^{th}-moment and sampling an edge from an almost uniform distribution) the graph constructions are similar or identical to the lower bound constructions in the original works. Our contribution is in the simplicity of the analysis.

4.1 Counting Subgraphs

Let $G = (V, E)$ be a graph and $H = (V_H, E_H)$ be a fixed graph with $|V_H| = k$. We denote the number of instances of H in G by $h_H(G)$. That is, $h_H(G)$ is the number of subgraphs $G' = (V', E')$ with $V' \subseteq V$, $E' \subseteq E$, and $|V'| = k$ such that G' is isomorphic to H.

▶ **Theorem 4.1.** *Let k be a fixed constant, $H = (V_H, E_H)$ a fixed graph on k vertices ($|V_H| = k$), and G' any graph on $n/2$ vertices. For any $\mu \leq \binom{n/2}{k}$, any algorithm $\mathcal{A}$ that distinguishes between graphs G on n vertices satisfying $h_H(G) = h_H(G')$ and $h_H(G) = h_H(G') + \mu$ requires $\Omega(n/\mu^{1/k})$ degree, neighbor, or pair queries in expectation.*

Proof. We apply Theorem 3.3 with $f = \text{disj}$, the disjointness function with input size $N = \Omega(n/\mu^{1/k})$. For fixed n and $x, y \in \{0,1\}^N$ we construct a graph $G = \mathcal{E}(x, y)$ on n vertices as follows. Take $V = \{v_1, v_2, \ldots, v_n\}$. Let ℓ be the smallest integer satisfying $\binom{\ell}{k} \geq \mu$ so that $\ell = O(\mu^{1/k})$. Take $N = n/2\ell$. We partition the first $n/2$ vertices into N sets of size ℓ which we denote $K_1, K_2, \ldots, K_N$. That is, $K_j = \{v_{j(\ell-1)+1}, v_{j(\ell-1)+2}, \ldots, v_{j\ell}\}$. The set of edges within K_j is determined by x_j and y_j. If $x_j = y_j = 1$, then K_j is a clique. Otherwise, K_j is a set of isolated vertices. Formally,

$$\text{for all } u, v \in K_j,\ (u, v) \in E \iff x_j = y_j = 1\ . \tag{1}$$

Edges are added to the remaining $n/2$ vertices of V (i.e., vertices $V_2 = \{v_{n/2+1}, \ldots, v_n\}$) so that the induced subgraph on V_2 is isomorphic to G'. Finally, let $g : \mathcal{G}_n \to \{0,1\}$ be the (partial) function defined by

$$g(G) = \begin{cases} 1 & \text{if } h_H(G) \leq h_H(G') \\ 0 & \text{if } h_H(G) \geq h_H(G') + \mu \end{cases}$$

We claim that $(\mathcal{E}, g)$ is an embedding of disj. To see this, first note that if $\text{disj}(x, y) = 1$, then the condition of Equation (1) is never satisfied. Thus, G is isomorphic to G', plus $n/2$ isolated vertices. In particular, $h_H(G) = h_H(G')$. On the other hand, if $\text{disj}(x, y) = 0$, then there exists $j \in [N]$ with $x_j = y_j = 1$. Thus K_j is a clique on ℓ vertices, implying that $h_H(K_j) \geq \binom{\ell}{k} \geq \mu$. Therefore, $h_H(G) \geq h_H(G') + h_H(K_j) \geq h_H(G') + \mu$, and the claim follows.

Finally, in order to apply Theorem 3.3, we must show that each degree, neighbor, or pair query can be simulated by Alice and Bob (who know x and y, respectively) using few bits. Let $u, v \in V$.

Degree query. Notice that if $u \notin K_1 \cup \cdots \cup K_N$, then Alice and Bob can compute $d(u)$ with no communication, as $d(u)$ does not depend on x or y. If $u \in K_j$, then Alice and Bob can compute $d(u)$ by exchanging x_j and y_j, which requires 2 bits.

Neighbor query. Again, if $u \notin K_1 \cup \cdots \cup K_n$, Alice and Bob can compute $\text{nbr}_i(u)$ without communication (by specifying some ordering on the edges of G' ahead of time). For $u \in K_j = \{v_{\ell(j-1)+1}, \ldots v_{\ell j}\}$, Alice and Bob can again compute $\text{nbr}_i(u)$ by exchanging x_j and y_j. To this end, if $x_j = 0$ or $y_j = 0$, then $\text{nbr}_i(u) = \varnothing$ for all i. If $x_j = y_j = 1$, then we can order the neighbors of $u = v_{\ell(j-1)+z}$ as follows: the i^{th} neighbor of u is $v_{\ell(j-1)+z+i}$, where the sum $z + i$ is computed modulo ℓ.

Pair query. The query $\text{pair}(u, v)$ depends only on x and y if $u, v \in K_j$ for some j. In this case, $\text{pair}(u, v) = 1$ if and only if $x_j = y_j = 1$. Thus Alice and Bob can simulate $\text{pair}(u, v)$ with 2 bits of communication.

Thus, all queries can be simulated using at most 2 bits of communication between Alice and Bob. Therefore, by Theorem 3.3, any algorithm that computes g requires $\Omega(R(\text{disj})/B) = \Omega(N/2) = \Omega(n/\ell) = \Omega(n/\mu^{1/k})$ queries, as desired. ◀

▶ **Corollary 4.2** (Theorem 3.2 in [18]). *Suppose $\mathcal{A}$ is an algorithm that gives a $(1+\varepsilon)$ multiplicative approximation to the number of edges in a graph using neighbor, degree, and pair queries. Specifically, for any $\varepsilon > 0$, on any input graph $G = (V, E)$ with $|V| = n$ and $|E| = m$, $\mathcal{A}$ outputs an estimate $\widehat{m}$ satisfying $\Pr(|\widehat{m} - m| < \varepsilon) \geq 2/3$. Then the expected query complexity of $\mathcal{A}$ is $\Omega(n/\sqrt{\varepsilon m})$.*

Proof. Apply Theorem 4.1 where H is a graph consisting of two vertices connected by a single edge. Take G' to be any graph on $n/2$ nodes with m edges, and take $\mu = 3\varepsilon m$. Observe that a $(1 \pm \varepsilon)$ multiplicative approximation to the number of edges in a graph distinguishes graphs with m edges from those with at least $(1 + 3\varepsilon)m$ edges for any $\varepsilon < 1/3$. ◀

▶ **Corollary 4.3** (cf. [9, 10]). *Suppose $\mathcal{A}$ is an algorithm that gives a $(1+\varepsilon)$ multiplicative approximation to the number of r-cliques in a graph using neighbor, degree, and pair queries. Specifically, for any $\varepsilon > 0$, on any input graph $G = (V, E)$ with $|V| = n$ containing C_r r-cliques, $\mathcal{A}$ outputs an estimate $\widehat{C_r}$ satisfying $\Pr(|\widehat{C_r} - C_r| < \varepsilon) \geq 2/3$. Then the expected query complexity of $\mathcal{A}$ is $\Omega(n/(\varepsilon C_r)^{1/r})$.*

▶ Remark 4.4. The lower bound of Corollary 4.2 is tight, as a matching upper bound is given in [18]. In the full version [12], we apply Theorem 3.3 to show that any $(2-\varepsilon)$ approximation to m requires $\Omega(n^2/m)$ queries if only degree queries are allowed. This fact was observed by Feige [14], who also showed that $O(n/\sqrt{m})$ degree queries are sufficient to obtain a 2-approximation of m.

The lower bound of Corollary 4.3 is tight for some ranges of the parameters n, m, and C_r, but not for the entire range (see [9, 10]). In Section 4.3, we apply Theorem 3.3 (and Corollary 4.3) to obtain a tight lower bound for approximating the number of triangles in a graph (C_3) over the entire range of parameters, thereby proving the lower bound of [9]. In the full version of this paper [12], we apply the same methodology to prove the lower bound of [10] for general r.

4.2 Sampling Edges

In this section, we prove a lower bound on the number of queries necessary to sample an edge in a graph $G = (V, E)$ from an "almost-uniform" distribution D over E. The lower bound we obtain – originally proven in [13] – is tight, as a matching upper bound is proven in [13]. Here, we use "almost uniform" in the sense of total variational distance:

▶ **Definition 4.5.** Let D and D' be probability distributions over a finite set X. Then the **total variational distance** between D and D' is defined by

$$\mathrm{dist}_{\mathrm{TV}}(D, D') = \frac{1}{2} \sum_{x \in X} |D(x) - D'(x)| .$$

For $\varepsilon > 0$, we say that D is **ε-close to uniform** if $\mathrm{dist}_{\mathrm{TV}}(D, U) \leq \varepsilon$ where U is the uniform distribution on X (i.e., $U(x) = 1/|X|$ for all $x \in X$).

▶ **Theorem 4.6** (cf. [13]). *Let $0 < \varepsilon < 1/3$. Suppose $\mathcal{A}$ is an algorithm that for any graph $G = (V, E)$ on n vertices and m edges returns an edge $e \in E$ sampled from a distribution D that is ε-close to uniform using neighbor, degree, and pair queries. Then $\mathcal{A}$ requires $\Omega(n/\sqrt{m})$ queries.*

Proof. We use the same embedding $\mathcal{E}$ of disj described in the proof of Theorem 4.1 where G' is any graph on m' edges, and $\ell = \sqrt{m'}$ so that $N = n/2\sqrt{m}$. Thus, if any K_j is a clique, the induced subgraph on $K = K_1 \cup \cdots \cup K_N$ contains at least $m/2$ edges in G (where m is the number of edges in G). We then take $g : \mathcal{G}_n \to \{0, 1\}$ to be the function whose value is 0 if and only if K contains an edge. It is clear that $(\mathcal{E}, g)$ is an embedding of disj. Thus, by Theorem 3.3 (and the proof of Theorem 4.1), any algorithm $\mathcal{A}'$ that computes g requires $\Omega(N) = \Omega(n/\sqrt{m})$ queries.

Let $\mathcal{A}$ be an algorithm as in the statement of the theorem. We will show that by invoking $\mathcal{A}$ $O(1)$ times, we can compute g. Thus, the lower bound on the number of queries for $\mathcal{A}$ follows from the lower bound on any algorithm $\mathcal{A}'$ computing g, as above. $\mathcal{A}'$ works as follows: repeat $\mathcal{A}$ 7 times to get edge samples $e_1, \ldots, e_7$. If at least one e_i satisfies $e_i \in K \times K$, return 0, otherwise return 1. We claim that this procedure computes g (on the range of $\mathcal{E}$). To see this, suppose $g(G) = 0$, i.e., at least one of the K_i is a clique so that $K \times K$ contains at least $m/2$ edges. Thus the fraction of edges in $K \times K$ is at least $1/2$, so each invocation of $\mathcal{A}$ must return an edge $e \in K \times K$ with probability at least $1/2 - 1/3 = 1/6$. Therefore, if $g(G) = 0$, the probability that algorithm $\mathcal{A}'$ returns 1 (i.e., that no edge $e \in K \times K$ is sampled) is at most $(1 - 1/6)^7 < 1/3$. On the other hand, if $g(G) = 1$, then the procedure will always return 1, as G contains no edges in $K \times K$. ◀

4.3 Counting Triangles

In this section, we prove lower lower bounds for approximately counting the number of triangles, C_3 in a graph. When combined with Corollary 4.3 (with $r = 3$), the main result in this section gives tight lower bounds for all ranges of the parameters n, m, and C_3. The lower bounds (and matching upper bounds) were originally described in [9].

▶ **Theorem 4.7** (cf. [9]). *Let G be a graph with n vertices, m edges, and C_3 triangles. Then any algorithm $\mathcal{A}$ that computes a multiplicative approximation for C_3 must perform $\Omega\left(\min\left\{m, \frac{m^{3/2}}{C_3}\right\}\right)$ degree, neighbor, or pair queries. This lower bound holds even if $\mathcal{A}$ is allowed to perform random edge queries (i.e., $\mathcal{A}$ may sample a random edge in G from a uniform distribution).*

We prove Theorem 4.7 by applying Theorem 3.3 with $f = \mathrm{int}_k$, where N is the size of the instance of int_k. For any choice of the parameters n, m, and C_3, we construct an embedding $(\mathcal{E}, g)$ of int_k such that if $\mathrm{int}_k(x, y) = 1$, then $\mathcal{E}(x, y)$ has (roughly) C_3 triangles; if $\mathrm{int}_k(x, y) = 0$, then $\mathcal{E}(x, y)$ is triangle-free.

Proof. Let n, m, and C_3 be given. In order to simplify our presentation, we assume that $C_3 \geq \frac{1}{2}\sqrt{m}$.[9] Let ℓ be a parameter (to be chosen later), and $N = \ell^2$ the size of an instance of int_k. We identify the set $\{0,1\}^N$ with $\{0,1\}^{\ell \times \ell}$, so that elements $x \in \{0,1\}^N$ are indexed by two parameters $x = (x_{ij})$ with $1 \leq i, j \leq \ell$.

For $x, y \in \{0,1\}^N$, we define $G = (V, E) = \mathcal{E}(x, y)$ as follows. We partition the vertex set V into 5 sets A, A', B, B', S each of size ℓ, along with an auxiliary set C of size $n - 5\ell$. The set C plays no role in our construction except to control the number of vertices in G. We denote $A = \{a_1, a_2, \ldots, a_\ell\}$, $A' = \{a'_1, a'_2, \ldots, a'_\ell\}$, and similarly for the remaining sets in the partition. The edge set E is constructed as follows:

- For all $a \in A$, $b \in B$ and $s \in S$, we have $(a, s), (b, s) \in E$.
- For all $i, j \in [\ell]$ we have

$$\begin{cases} (a_i, b_j), (a'_j, b'_i) \in E & \text{if } x_{ij} = y_{ij} = 1 \\ (a_i, a'_j), (b_j, b'_i) \in E & \text{otherwise.} \end{cases}$$

Define the partial function $g : \mathcal{G}_n \to \{0,1\}$ by $g(G) = 0$ if $C_3(G) = 0$, and $g(G) = 1$ if $C_3(G) \geq k\ell$. In Appendix B (Claim B.1), we prove that $(\mathcal{E}, g)$ is an embedding of int_k. To apply Theorem 3.3, we must show that each allowable query can be simulated by Alice and Bob holding x and y, respectively using $O(1)$ bits of communication. A proof of this fact is deferred to the appendix (Claim B.2).

Since all queries can be simulated using $O(1)$ bits of communication between Alice and Bob and $N = \ell^2$, Theorem 3.3 (together with communication lower bound for int_k, Corollary 2.7) implies that computing g requires $\Omega(\ell^2/k)$ queries. For a k-intersecting instance (i.e., $\text{int}_k(x, y) = 1$), we have $m = 4\ell^2$ and $C_3(G) \geq k\ell$. Thus, setting $\ell = \frac{1}{2}\sqrt{m}$ and $k = C_3/\ell$, we obtain the desired result when $2C_3/\sqrt{m} \geq 1$. In the case where $C_3 < \frac{1}{2}\sqrt{m}$, we may take $k = 1$ and modify the construction so that $|S| = C_3$. ◀

▶ Remark 4.8. Combined with Corollary 4.3 with $r = 3$, Theorem 4.7 gives a lower bound of $\Omega\left(\min\left\{m, m^{3/2}/C_3\right\} + n/C_3^{1/3}\right)$ queries for any algorithm that obtains a multiplicative approximation to C_3 in the general graph model with no random edge samples. This lower bound is tight by the matching upper bound in [9]. When random edge samples are allowed, however, the lower bound construction of Theorem 4.1 (hence Corollary 4.3) does not yield the same lower bound. To see this, note that a random edge sample will be in K_j with probability ε, so only $O(1/\varepsilon)$ such samples are sufficient distinguish $\text{disj}(x, y) = 1$ from 0-instances with constant probability. When random edge samples are allowed, we conjecture that the lower bound of Theorem 4.7 is tight for the entire range of n, m, and C_3.

4.4 Computing Edge Connectivity

In this section, we consider the problem of estimating the edge connectivity of a graph. Recall that a graph $G = (V, E)$ is *k-(edge)-connected* if at least k edges must be removed from G in order to disconnect it. Equivalently, G is k-connected if for every $u, v \in V$, there are at least k edge-disjoint paths between u and v. We prove the following lower bound for determining the connectivity of a graph G.

▶ **Theorem 4.9.** *For $k \geq 1$, let G be a graph with n vertices and $m \geq 2kn$ edges. Then any algorithm $\mathcal{A}$ that distinguishes between the case where G is k-connected and G is disconnected requires $\Omega(m/k)$ degree, neighbor, or pair queries. This lower bound holds even if $\mathcal{A}$ is allowed to perform random edge queries.*

[9] At the end of the proof we will discuss how to avoid this unnecessary assumption.

Proof. The proof uses a similar construction and analysis to that of Theorem 4.7. Again, we describe an embedding $(\mathcal{E}, g)$ of int_k, for which we provide full details. The correctness of the embedding and simulation arguments are omitted, as they are nearly identical to those in the proof of Theorem 4.7.

Let $\ell \geq 2k$ be parameter to be chosen later and $N = \ell^2$. Again we identify $\{0,1\}^N = \{(x_{ij}) \mid 1 \leq i, j \leq \ell\}$. For $x, y \in \{0,1\}^N$, the graph $\mathcal{E}(x,y) = (V, E)$ is constructed as follows. We partition V into 5 sets, $V = A \cup A' \cup B \cup B' \cup C$ where $|A| = |A'| = |B| = |B'| = \ell$, and $|C| = n - 4\ell$. Each $v \in C$ is connected to k distinct vertices in A arbitrarily so that $d(v) = k$. We then construct edges between A, A', B, and B' according to the following rule:

$$\begin{cases} (a_i, b'_j), (b_i, a'_j) \in E & \text{if } x_{ij} = y_{ij} = 1 \\ (a_i, a'_j), (b_i, b'_j) \in E & \text{otherwise.} \end{cases} \tag{2}$$

We define the partial function $g : \mathcal{G}_n \to \{0,1\}$ by $g(G) = 1$ if G is k-connected, and 0 if G is disconnected.

We claim that $(\mathcal{E}, g)$ is an embedding of int_k () – a proof appears in Appendix C. As in the proof of Theorem 4.7, every degree, neighbor, pair, or random edge query can be simulated by Alice and Bob using at most 2 bits of communication. Therefore, by Theorem 3.3 and Corollary 2.7, $\mathcal{A}$ requires $\Omega(N/k) = \Omega(\ell^2/k)$ queries. The construction above satisfies $m = 2\ell^2 + k(n - 4\ell)$ so taking $\ell = k + \sqrt{k^2 + (m - kn)/2} = \Theta(\sqrt{m})$ gives the desired result. ◀

5 Discussion

In this paper, we presented a new technique for proving query lower bounds for graph parameter estimation problems. Here, we conclude with some open questions and suggestions for further work.

The power of random edge queries. In [2], Aliakbarpour et al. consider a graph query model that allows uniform random edge samples as one of its basic queries. This model is strictly stronger than the "general graph" query model that allows only degree, neighbor, and pair queries: [2] provides an upper bound for counting the number of star sugraphs in a graph that beats the lower bound described in [20] for the general graph model. In Theorems 4.7 and 4.9, our lower bounds apply to both the general graph model, as well as the stronger model with uniform random edge samples. In the constructions described in the proofs of Theorems 4.7 and 4.9, edge samples do not afford more computational power, essentially because the degree sequence of the constructions is fixed. In particular, each random edge sample can be simulated by sampling a vertex with probability proportional to its (known) degree, and using a neighbor query to sample a random incident edge. Indeed, sampling edges from a uniform distribution is equivalent to sampling vertices with probability proportional to their degrees.

The construction used for lower bound of Theorem 4.1 cannot give a lower bound better than $\Omega(\min\{m/\mu^{2/k}, n/\mu^{1/k}\})$, as a clique K_j in that construction contains $\mu^{2/k}$ edges. In particular, the lower bound for estimating m (Corollary 4.2) becomes only $\Omega(1/\varepsilon)$ if random edge samples are used. An upper bound of $O(n^{1/3})$ for estimating m with random edge (and vertex) samples is implied by the algorithm of Motwani et al. [27]. The authors also prove a lower bound of $\Omega(n^{1/3})$, although the construction only holds for $m = O(n^{2/3})$. We believe it is an interesting problem to characterize the complexity of estimating m in the general graph model with random edge samples the over the full range of m.

In general, we would like to better understand the power uniform random edge samples in the general graph model. We conjecture that the lower bound of Theorem 4.7 is tight over the entire range of the parameters. The algorithm of Eden et al. [9] proves that the lower bound is tight for $\min\left\{m, m^{3/2}/C_3\right\} = \Omega(n/C_3^{1/3})$ even without edge samples. Thus, edge samples may only help in the regime where $n/C_3^{1/3} = \omega(\min\left\{m, m^{3/2}/C_3\right\})$.

▶ Question 5.1. For what graph parameter estimation problems do random edge samples help?

Property testing lower bounds. The graph query access models we consider were initially proposed in the context of property testing [16, 17, 28, 25]. In graph property testing with the general graph model [25], the goal is to distinguish graphs that satisfy some property P from those that are far from satisfying P in the sense that an ε-fraction of edges of the graph must be modified in order to the graph to satisfy P. In this model, our constructions imply property testing lower bounds, at least for some range of m. For example, we state a consequence of Theorem 4.1 for testing the property of triangle-freeness (i.e., that $C_3(G) = 0$).

▶ **Corollary 5.2.** *Any property testing algorithm for triangle-freeness in the general graph model requires $\Omega(n/\sqrt{\varepsilon m})$ queries.*

Corollary 5.2 follows from the construction in Theorem 4.1 by taking G' to be any triangle-free graph on $n/2$ vertices and m edges, and $\ell = \sqrt{\varepsilon m}$. This way each (potential) clique K_j on ℓ vertices will contain roughly εm edges. Further, by Turán's theorem [33, 1], at least (roughly) $\varepsilon/3m$ edges must be removed from G (in particular from K_j) in order to make G triangle-free in the case where K_j is a clique. The lower bound of Corollary 5.2 matches the known lower bound due to Alon et al. [3] in the regime where the average degree $d = 2m/n$ satisfies $d = O(n^{1/3})$. In the range $d = \omega(n^{1/3})$, the lower bounds in [3] are strictly stronger.[10]

In the dense graph model [16], only pair queries are allowed, but the distances between graphs are normalized by n^2 (rather than $|E|$ as in the general graph model). Thus every graph with $m = o(n^2)$ is ε-close to the graph with no edges. In this case, the types of embeddings we present in this paper cannot yield lower bounds that are better than $\Omega(1/\varepsilon)$. Specifically, in all embeddings of disjointness we consider, each edge in $\mathcal{E}(x, y)$ depends on a single bit of $x, y \in \{0, 1\}^N$. However, the value of $\mathrm{disj}(x, y)$ can vary by changing a single bit of x or y. In order for Theorem 3.3 to give property testing lower bounds via an embedding of disj, changing a single bit of x or y must change εn^2 edges in $\mathcal{E}(x, y)$. Thus, to obtain stronger lower bounds (e.g., lower bounds that grow as a function of n), either different communication primitives must be considered, or the embedding $\mathcal{E}$ must be more complicated (using some nontrivial encoding of x and y).

▶ Question 5.3. Can Theorem 3.3 be applied to obtain any nontrivial (i.e., $\omega(1/\varepsilon)$) lower bound for any "natural" graph problem in the dense graph property testing model?

[10] Alon et al. [3] prove that a lower bound of $\Omega(n^{1/3})$ holds even for all $d = O(n^{1-\nu(n)})$ for some function $\nu(n) = o(1)$. This is in contrast for the case $d = \Omega(n)$, where $O(f(\varepsilon))$ queries are sufficient for some function f.

References

1. Martin Aigner and Günter M Ziegler. *Proofs from the Book*. Springer-Verlag Berlin Heidelberg, 4th edition, 2010. doi:10.1007/978-3-642-00856-6.
2. Maryam Aliakbarpour, Amartya Shankha Biswas, Themistoklis Gouleakis, John Peebles, Ronitt Rubinfeld, and Anak Yodpinyanee. Sublinear-time algorithms for counting star subgraphs with applications to join selectivity estimation. *CoRR*, abs/1601.04233, 2016. arXiv:1601.04233.
3. Noga Alon, Tali Kaufman, Michael Krivelevich, and Dana Ron. Testing triangle-freeness in general graphs. *SIAM Journal on Discrete Mathematics*, 22(2):786–819, 2008.
4. 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 Theory of computing*, pages 20–29. ACM, 1996.
5. Ziv Bar-Yossef, T.S. Jayram, Ravi Kumar, and D. Sivakumar. An information statistics approach to data stream and communication complexity. *Journal of Computer and System Sciences*, 68(4):702–732, 2004. Special Issue on FOCS 2002. doi:10.1016/j.jcss.2003.11.006.
6. Suman K. Bera and Amit Chakrabarti. Towards Tighter Space Bounds for Counting Triangles and Other Substructures in Graph Streams. In Heribert Vollmer and Brigitte Vallée, editors, *34th Symposium on Theoretical Aspects of Computer Science (STACS 2017)*, volume 66 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 11:1–11:14, Dagstuhl, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.STACS.2017.11.
7. Eric Blais, Joshua Brody, and Kevin Matulef. Property testing lower bounds via communication complexity. *computational complexity*, 21(2):311–358, Jun 2012. doi:10.1007/s00037-012-0040-x.
8. Shahar Dobzinski, Noam Nisan, and Michael Schapira. Approximation algorithms for combinatorial auctions with complement-free bidders. In *Proceedings of the Thirty-seventh Annual ACM Symposium on Theory of Computing*, STOC '05, pages 610–618, New York, NY, USA, 2005. ACM. doi:10.1145/1060590.1060681.
9. T. Eden, A. Levi, D. Ron, and C. Seshadhri. Approximately counting triangles in sublinear time. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 614–633, Oct 2015. doi:10.1109/FOCS.2015.44.
10. Talya Eden, Dana Ron, and C. Seshadhri. On approximating the number of k-cliques in sublinear time. *CoRR*, abs/1707.04858, 2017. arXiv:1707.04858.
11. 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. Full version available at https://arxiv.org/abs/1604.03661. doi:10.4230/LIPIcs.ICALP.2017.7.
12. Talya Eden and Will Rosenbaum. Lower bounds for approximating graph parameters via communication complexity. Full version of this paper, 2017. URL: https://arxiv.org/abs/1709.04262.
13. Talya Eden and Will Rosenbaum. On Sampling Edges Almost Uniformly. In Raimund Seidel, editor, *1st Symposium on Simplicity in Algorithms (SOSA 2018)*, volume 61 of *Open-Access Series in Informatics (OASIcs)*, pages 7:1–7:9, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. Full version available at https://arxiv.org/abs/1706.09748. doi:10.4230/OASIcs.SOSA.2018.7.
14. Uri Feige. On sums of independent random variables with unbounded variance and estimating the average degree in a graph. *SIAM J. Comput.*, 35(4):964–984, 2006. doi:10.1137/S0097539704447304.

15 Oded Goldreich. On the communication complexity methodology for proving lower bounds on the query complexity of property testing. *Electronic Colloquium on Computational Complexity (ECCC)*, 20:73, 2013. URL: `http://eccc.hpi-web.de/report/2013/073`.

16 Oded Goldreich, Shari Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM (JACM)*, 45(4):653–750, 1998.

17 Oded Goldreich and Dana Ron. Property testing in bounded degree graphs. *Algorithmica*, pages 302–343, 2002.

18 Oded Goldreich and Dana Ron. Approximating average parameters of graphs. *Random Structures & Algorithms*, 32(4):473–493, 2008. `doi:10.1002/rsa.20203`.

19 Yannai A. Gonczarowski, Noam Nisan, Rafail Ostrovsky, and Will Rosenbaum. A stable marriage requires communication. In *Proceedings of the Twenty-sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '15, pages 1003–1017, Philadelphia, PA, USA, 2015. Society for Industrial and Applied Mathematics. URL: `http://dl.acm.org/citation.cfm?id=2722129.2722197`.

20 Mira Gonen, Dana Ron, and Yuval Shavitt. Counting stars and other small subgraphs in sublinear-time. *SIAM Journal on Discrete Mathematics*, 25(3):1365–1411, 2011.

21 J. Hromkovič. *Communication Complexity and Parallel Computing*. Texts in Theoretical Computer Science. An EATCS Series. Springer Berlin Heidelberg, 2013.

22 Bala Kalyanasundaram and Georg Schintger. The probabilistic communication complexity of set intersection. *SIAM Journal on Discrete Mathematics*, 5(4):545–557, 1992.

23 Bala Kalyanasundaram and Georg Schnitger. *Communication Complexity and Lower Bounds for Sequential Computation*, pages 253–268. Vieweg+Teubner Verlag, Wiesbaden, 1992. `doi:10.1007/978-3-322-95233-2_15`.

24 Mauricio Karchmer and Avi Wigderson. Monotone circuits for connectivity require super-logarithmic depth. *SIAM Journal on Discrete Mathematics*, 3(2):255–265, 1990. `doi:10.1137/0403021`.

25 Tali Kaufman, Michael Krivelevich, and Dana Ron. Tight bounds for testing bipartiteness in general graphs. *SIAM Journal on computing*, 33(6):1441–1483, 2004.

26 E. Kushilevitz and N. Nisan. *Communication Complexity*. Cambridge University Press, 2006.

27 Rajeev Motwani, Rina Panigrahy, and Ying Xu. Estimating sum by weighted sampling. In *ICALP*, volume 4596, pages 53–64. Springer, 2007.

28 Michal Parnas and Dana Ron. Testing the diameter of graphs. *Random Structures & Algorithms*, 20(2):165–183, 2002. `doi:10.1002/rsa.10013`.

29 Ramamohan Paturi and Janos Simon. Probabilistic communication complexity. *Journal of Computer and System Sciences*, 33(1):106–123, 1986.

30 Alexander A. Razborov. On the distributional complexity of disjointness. *Theoretical Computer Science*, 106(2):385–390, 1992.

31 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. *SIAM Journal on Computing*, 41(5):1235–1265, 2012. `doi:10.1137/11085178X`.

32 Jacob Steinhardt, Gregory Valiant, and Stefan Wager. Memory, communication, and statistical queries. In Vitaly Feldman, Alexander Rakhlin, and Ohad Shamir, editors, *29th Annual Conference on Learning Theory*, volume 49 of *Proceedings of Machine Learning Research*, pages 1490–1516, Columbia University, New York, New York, USA, 23–26 Jun 2016. PMLR. URL: `http://proceedings.mlr.press/v49/steinhardt16.html`.

33 Paul Turán. On an extremal problem in graph theory. *Matematikai és Fizikai Lapok*, 48:436–452, 1941.

34 Andrew Chi-Chih Yao. Some complexity questions related to distributive computing(preliminary report). In *Proceedings of the Eleventh Annual ACM Symposium on Theory of Computing*, STOC '79, pages 209–213, New York, NY, USA, 1979. ACM. doi:10.1145/800135.804414.

A k-Intersection Lower Bound

Proof of Corollary 2.7. The argument is by simulation. Specifically, we will show that any efficient protocol for int_k yields an efficient protocol for disj. Suppose Π is a protocol for the promise problem of the corollary with $\text{cost}(\Pi) = B$. For $x, y \in \{0,1\}^{N/k}$, let $x^k, y^k \in \{0,1\}^N$ denote the concatenation of x and y (respectively) repeated k times. Observe that if x and y satisfy the unique intersection promise, then x^k and y^k satisfy the k-intersection promise. Further, $\text{int}_k(x^k, y^k) = 0$ if and only if $\text{disj}(x, y) = 1$. Since Π computes int_k for all $x', y' \in \{0,1\}^N$ satisfying the k-intersection promise, $\Pi(x^k, y^k)$ computes $\neg\,\text{disj}$ on input x, y. Therefore, by Theorem 2.5, $\text{cost}(\Pi) = \Omega(N/k)$, which gives the desired result. ◀

B Theorem 4.7 Details

In this appendix, we provide details to complete the proof of Theorem 4.7.

▶ **Claim B.1.** *Let $\mathcal{E}$ be defined as in Section 4.3, and define the partial function $g : \mathcal{G} \to \{0,1\}$ by*

$$g(G) = \begin{cases} 0 & \text{if } C_3(G) = 0 \\ 1 & \text{if } C_3(G) \geq k\ell. \end{cases}$$

Then $(\mathcal{E}, g)$ is an embedding of int_k.

Proof. To see the claim is true, first consider the case where $\text{int}_k(x, y) = 0$ – i.e., x and y are disjoint. In this case, all edges in $\mathcal{E}(x, y)$ are between A and S, A and A', B and S, or B and B'. Therefore there are no edges between $S \cup A \cup B$ and $A' \cup B' \cup C$, implying that G is bipartite, hence triangle-free. On the other hand, if $\text{int}_k(x, y) = 1$, then for each of the (at least k) pairs $(i, j) \in [\ell]^2$ satisfying $x_{ij} = y_{ij} = 1$, we have $(a_i, b_j) \in E$. Therefore, for each $s \in S$, the edges $(a_i, b_j), (b_j, s), (s, a_i) \in E$ form a triangle. Since $|S| = \ell$, this implies that $\mathcal{E}(x, y)$ contains at least $k\ell$ triangles. ◀

▶ **Claim B.2.** *For the embedding $(\mathcal{E}, g)$ described in Section 4.3, any degree, neighbor, or pair query, and a random edge sampling can be simulated using $O(1)$ bits of communication between Alice and Bob.*

Proof. We consider each allowable query separately. For any $u, v \in V$, the queries can be simulated as follows.

Degree query. $d(u)$ is independent of x and y: if $u \in S \cup A \cup B$, then $d(u) = 2\ell$; if $u \in A' \cup B'$, then $d(u) = \ell$; if $u \in C$, then $d(u) = 0$. Thus Alice and Bob can simulate any degree query without communication.

Neighbor query. For $a_i \in A$, we label a_i's incident edges so that a_i's j^{th} neighbor is either b_j (if $x_{ij} = y_{ij} = 1$) or a'_j otherwise. Edges incident to A', B, and B' are labeled similarly. Thus, Alice and Bob can answer queries of the form $\text{nbr}_i(u)$ for $v \in A \cup A' \cup B \cup B'$ with $j \leq \ell$ by exchanging x_{ij} and y_{ij} using 2 bits of communication. All other neighbor queries can be answered without communication.

Pair query. Alice and Bob can answer pair queries of the form $\text{pair}(a_i, a'_j)$, $\text{pair}(a_i, b_j)$, and $\text{pair}(b_j, b'_i)$ by exchanging x_{ij} and y_{ij}. All other queries can be answered without communication. Again the communication cost is 2 bits.

Uniform edge sample. Alice and Bob can sample a random edge from a uniform distribution using their shared public randomness and the fact that each node in $\mathcal{E}(x, y)$ has the same degree independent of x and y. To this end, Alice and Bob sample $e = (u, v)$ by first sampling a vertex $v \in V$ where each node is chosen with probability proportional to its degree, $d(v)$. Alice and Bob then choose a random number $i \in [d(v)]$ uniformly at random, and sample the edge $e = (v, u)$ where $u = \text{nbr}_i(v)$ at a communication cost of (at most) 2 bits. Note that $e = (v, u)$ is sampled with probability

$$\Pr(e = (v, u) \text{ is sampled}) = \frac{d(v)}{\sum_{w \in V} d(w)} \cdot \frac{1}{d(v)} + \frac{d(u)}{\sum_{w \in V} d(w)} \frac{1}{d(u)} = \frac{1}{m},$$

so that edges are indeed sampled according to a uniform distribution. ◀

C Theorem 4.9 Details

▶ **Claim C.1.** *Let $\mathcal{E}$ and g be as described in Section 4.4. Then $(\mathcal{E}, g)$ is an embedding of* int_k *(where we assume the promise that $\sum_{i,j} x_{ij} y_{ij} \in \{0, k\}$).*

Proof. In the case where $\text{int}_k(x, y) = 0$, there are no edges between $A \cup A' \cup C$ and $B \cup B'$, hence $\mathcal{E}(x, y)$ is disconnected. If $\text{int}_k(x, y) = 1$, we must show that $\mathcal{E}(x, y)$ is k-connected. We will show that there are at least k edge disjoint paths between any pair of vertices in $\mathcal{E}(x, y)$. We consider the following cases separately.

Case 1: $u, v \in A$ (or symmetrically, $u, v \in A', B$, or B'). From the definition of $\mathcal{E}$ (Equation (2)) and the promise for int_k, there are at most k pairs $(a_i, a'_j) \in A \times A'$ that are *not* contained in E. Since $\ell \geq 2k$, this implies that $u, v \in A$ have at least $\ell - k \geq k$ common neighbors in A'. In particular, there are at least this many edge disjoint paths (of length 2) between u and v.

Case 2: $u \in A$, $v \in A'$ (or symmetrically, $u \in B$, $v \in B'$). As before, v has at least k distinct neighbors, $u_1, \ldots, u_k \in A$. Further, by the analysis in Case 1, each u_i has at least k common neighbors with u. Therefore there exists a matching $(u_1, v_1), \ldots, (u_k, v_k) \in E$.[11] The paths $(u, v_i), (v_i, u_i), (u_i, v)$ for $i = 1, \ldots, k$ are then edge disjoint. (Note that it may be the case that $u_i = u$, in which case we take (u, v) to be one of the matching edges and take this edge to be the corresponding path between u and v.)

Case 3: $u \in A$, $v \in B'$ (or symmetrically $u \in A'$, $v \in B$). Let $(u_1, v_1), \ldots, (u_k, v_k) \in A' \times B$ be the k edges between A' and B. Take $U = \{u_1, \ldots, u_k\}$ (respectively, $V = \{v_1, \ldots, v_k\}$) to be the multiset of endpoints of the edges between A' and B in A' (respectively B). It suffices to show that are edge-disjoint paths from u to each u_i (with multiplicity) – the analogous result for v and v_i is identical. Let $u'_1, u'_2, \ldots, u'_k$ be distinct neighbors of u. Then, as in Case 1, each u_i and u'_i have at least k common neighbors. By choosing one such common neighbor, u''_i for each i (greedily) we can form k edge disjoint paths $(u, u'_i), (u'_i, u''_i), (u''_i, u_i)$.

Case 4: $u \in A$, $v \in B$ (or symmetrically, $u \in A'$, $v \in B'$). Let $(u_1, v_1), \ldots, (u_k, v_k)$ be as in Case 3. As in Case 3, there are k edge-disjoint paths in $A \cup A'$ from u to the u_i. Further, there are k edge-disjoint paths from v to the v_j as in Case 2.

[11] Such a matching can be found by greedily choosing common neighbors of u and u_1, u_2, etc.

Case 5: $u \in C$. Let $u_1, \ldots, u_k$ be the neighbors of u in A. It suffices to show that there are edge disjoint paths from each u_i to v. The cases $v \in A, A', B, B'$ are analogous to arguments in Cases 1–4. If $v \in C$, let $v_1, \ldots, v_k$ be the neighbors of v in A. Since each pair u_i, v_i share k common neighbors in A', we can assign a unique neighbor w_i to each such pair so that $(u, u_i), (u_i, w_i), (w_i, v_i), (v_i, v)$ for $i = 1, \ldots, k$ are edge disjoint paths. ◀

Communication Complexity of Correlated Equilibrium with Small Support

Anat Ganor[1]
Tel Aviv University, Tel Aviv, Israel
anat.ganor@gmail.com

Karthik C. S.[2]
Weizmann Institute of Science, Rehovot, Israel
karthik.srikanta@weizmann.ac.il

Abstract

We define a two-player $N \times N$ game called the 2-cycle game, that has a unique pure Nash equilibrium which is also the only correlated equilibrium of the game. In this game, every $1/\mathrm{poly}(N)$-approximate correlated equilibrium is concentrated on the pure Nash equilibrium. We show that the randomized communication complexity of finding any $1/\mathrm{poly}(N)$-approximate correlated equilibrium of the game is $\Omega(N)$. For small approximation values, our lower bound answers an open question of Babichenko and Rubinstein (STOC 2017).

2012 ACM Subject Classification Theory of computation → Communication complexity, Theory of computation → Exact and approximate computation of equilibria

Keywords and phrases Correlated equilibrium, Nash equilibrium, Communication complexity

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.12

Acknowledgements We thank Yakov Babichenko, Moni Naor, Noam Nisan, Aviad Rubinstein, Amir Shpilka and Eylon Yogev for very helpful conversations and the anonymous reviewers for their valuable feedback on earlier versions of this manuscript.

1 Introduction

> If there is intelligent life on other planets, in a majority of them, they would have discovered correlated equilibrium before Nash equilibrium.
>
> *Roger Myerson*

One of the most famous solution concepts in game theory is Nash equilibrium [28]. Roughly speaking, a Nash equilibrium is a set of mixed strategies, one per player, from which no player has an incentive to deviate. A well-studied computational problem in algorithmic game theory is that of finding a Nash equilibrium of a (non-cooperative) game. Since finding a Nash equilibrium is considered hard (in particular, it is a PPAD-complete problem), researchers studied the problem of finding an approximate Nash equilibrium, where intuitively, no player can benefit much by deviating from his mixed strategy. The complexity of finding an approximate Nash equilibrium has been studied in several models of computation, including computational complexity, query complexity and communication

[1] The research leading to these results has received funding from the Israel Science Foundation (grant number 552/16) and the I-CORE Program of the planning and budgeting committee and The Israel Science Foundation (grant number 4/11).
[2] This work was supported by Irit Dinur's ERC-CoG grant 772839 and ISF-UGC grant 1399/14.

complexity. For surveys on algorithmic game theory in general and equilibria in particular see for example [29, 32, 13, 34].

When the players reach an equilibrium they learn to predict correctly the actions of the other players. To better understand how players might learn the actions and payoffs of other players, there is an extensive study of learning dynamics and their convergence to equilibria, see for example [22, 37, 19]. One natural class of dynamics in which approximate equilibria concepts are studied is that of uncoupled dynamics [17, 18], where each player knows his own utilities and not those of the other players. The rate of convergence of uncoupled dynamics to an approximate equilibrium is closely related to the communication complexity of finding the approximate equilibrium [8].

Communication complexity is a central model in complexity theory that has been extensively studied. In the two-player randomized model of Yao [36], each player gets an input and their goal is to solve a communication task that depends on both inputs. The players can use both common and private random coins and are allowed to err with some small probability. The communication complexity of a protocol is the total number of bits communicated by the two players. The communication complexity of a communication task is the minimum number of bits that the players need to communicate in order to solve the task with high probability, where the minimum is taken over all protocols. For surveys on communication complexity see for example [25, 26, 33].

An important generalization of Nash equilibrium is correlated equilibrium [1, 2]. Whereas in a Nash equilibrium the players choose their strategies independently, in a correlated equilibrium the players can coordinate their decisions, choosing a joint strategy. There are two notions of correlated equilibrium which we call *correlated equilibrium* (CE) and *rule correlated equilibrium* (RCE)[3]. In a CE no player can benefit from replacing one action with another, whereas in a RCE no player can benefit from simultaneously replacing every action with another action (using a switching rule). While the above two notions are equivalent, approximate CE and approximate RCE are not equivalent, but are closely related.

The communication task of finding an approximate (rule) correlated equilibrium is as follows. The actions sets and the approximation value are known to both players. Each player gets a utility function that specify her payoffs for every pair of actions (given as a truth table). At the end of the communication both players should know the same correlated mixed strategy which is an approximate (rule) correlated equilibrium.

In the multi-party setting, [16, 30, 21] showed protocols for finding an exact CE of N-player binary action games with $\text{poly}(N)$ bits of communication (note that the input size per player is 2^N). In the two-player setting, every $N \times N$ game has a trivial $1/N$-approximate CE (the uniform distribution over all pairs of actions, which can be found with zero communication). However, there is no trivial approximate RCE, even for constant approximation values. Babichenko and Rubinstein [6] raised the following questions:

> *Does a* $\text{polylog}(N)$ *communication protocol for finding an approximate* RCE *of two-player* $N \times N$ *games exist? Is there a* $\text{poly}(N)$ *communication complexity lower bound?*

1.1 Our Main Result

We show a communication complexity lower bound for finding a $1/\text{poly}(N)$-approximate CE of a two-player $N \times N$ game that we call the 2-cycle game. Since every approximate RCE is an approximate CE, the same lower bound holds for RCE.

[3] In most literature, both notions of correlated equilibrium are referred to by the same name. In this paper, we choose to distinguish between them.

▶ **Theorem 1.** *Let $n \geq 3$ be an odd integer, let $N = 2n$ and $\varepsilon \leq \frac{1}{4N^3}$. Then, every randomized communication protocol for finding an ε-approximate correlated equilibrium of the 2-cycle $N \times N$ game, with error probability at most $\frac{1}{3}$, has communication complexity at least $\Omega(N)$.*

As far as we know, there were no communication complexity lower bounds for (exact) CE or RCE of two-player games prior to this work. Note that Theorem 1 implies a lower bound of $\Omega(N)$ on the number of queries to the utility functions, for any randomized query algorithm that finds a $1/\mathrm{poly}(N)$-approximate CE (or RCE) of the 2-cycle $N \times N$ game, with probability at least $2/3$.

After the first version of this paper was published online two similar results were proved. Babichenko [4] showed that any $1/\mathrm{poly}(N)$-approximate RCE in the generalized matching-pennies game reveals the entire input of one of the players, thus proved the same lower bound as in Theorem 1. Ko and Schvartzman [24] independently showed that for any $\Omega(1/N) < \varepsilon < 1/10$, the communication complexity of finding an ε-approximate RCE is $\Omega(\varepsilon^{-1/2} \log N)$. In our opinion, the main difference between our lower bound and the above results is that their games have no approximate RCE with succinct representation, while our game has a unique pure Nash equilibrium (which is also a RCE with one pair of actions in its support). We will elaborate on the importance of proving lower bounds for equilibria with succinct representations later in this section.

It remains a very interesting open problem to determine the communication complexity of finding a *constant*-approximate RCE of two-player games. Currently, there is an exponential gap between the best known lower and upper bounds on the communication complexity of finding a constant-approximate RCE of two-player $N \times N$ games, where the best known lower bound is logarithmic in N [24].

In a recent breakthrough, Babichenko and Rubinstein [6] proved the first non-trivial lower bound on the randomized communication complexity of finding an approximate Nash equilibrium. More precisely, they proved a lower bound of $\Omega\left(N^{\varepsilon_0}\right)$ on the randomized communication complexity of finding an ε-approximate Nash equilibrium of a two-player $N \times N$ game, for every $\varepsilon \leq \varepsilon_0$, where ε_0 is some small constant. Theorem 1 implies a randomized communication complexity lower bound of $\Omega(N)$ for finding a $1/\mathrm{poly}(N)$-approximate Nash equilibrium of the 2-cycle game[4]. This is a slightly stronger lower bound but for much smaller approximation values. Our proof is more simple and straightforward, as it does not go through several intermediate problems.

The 2-cycle game is a very simple game, in the sense that it is a win-lose, sparse game, in which each player has a unique best response to every action. For the class of win-lose, sparse games, our lower bound is tight up to logarithmic factors, as a player can send his entire utility matrix using $O(N \log N)$ bits of communication. However, our lower bound does not hold for much larger approximation values, since there are examples of approximate equilibria of the 2-cycle game for larger approximation values, that can be found with small amount of communication (see Appendix A for details).

Correlated Equilibrium with Succinct Representation

In the communication model, for a problem to be meaningful, we would like the output size to be much smaller than the number of bits needed to solve it. Specifically, for the problem of finding an approximate RCE (or CE) in $N \times N$ games, we would like the output to have a

[4] We are able to improve the approximation parameter from $1/4N^3$ in Theorem 1 above to $1/16N^2$ in the case of approximate Nash equilibrium.

succinct representation of size polylog(N) bits. A natural notion of succinct representation is the support size of an equilibrium, thus we could define the communication problem associated with finding a RCE to be finding an approximate RCE of polylog(N) support size.

For $1/\text{poly}(N)$ approximation values, there are two-player $N \times N$ games for which every approximate RCE is of poly(N) support size. In contrast, Babichenko, Barman and Peretz [5] showed that every two-player game has a constant-approximate RCE of support size $O(\log N)$. Therefore, finding an approximate RCE of polylog(N) support size is a *total* search problem (i.e., a solution always exists) for constant approximation values[5], but is not total for $1/\text{poly}(N)$ approximation values.

As a step before understanding the total problem of finding a constant-approximate RCE, we consider *promise* problems, where we are guaranteed to have a RCE with a small support. Our lower bound for finding an approximate RCE implies that even for games in which we are promised to have a RCE with a small support, finding an approximate RCE remains hard.

1.2 Sampling from a Correlated Equilibrium

We introduce another natural communication task in the context of joint strategies, which is the task of sampling from a CE. Intuitively, in the task of sampling from a CE (or RCE)[6], the players are required to output each pair of actions with probability that is close to the probability of this pair of actions under some CE. More formally, at the end of the communication, each player outputs a *single* action, such that the distribution of the protocol on pairs of actions is close (say, Δ-close in ℓ_1 distance, for some small Δ) to some joint distribution which is a CE of the game. The above problem was suggested by Moni Naor [27].

The problem of finding a CE in the communication model requires that both players know at the end of the communication the entire joint strategy, which might be large. We believe that the easier task of sampling from a CE is interesting in real-life scenarios, since by sampling from an equilibrium of the game the players can act according to that equilibrium. Sampling communication tasks were studied in many different variants in different contexts, such as compression of randomized protocols, simulation of randomized protocols, agreement distillation, sketching algorithms, approximation algorithms based on rounding linear programming relaxations, the study of parallel repetition and cryptography.

When a game has a CE with a small support size, by sampling from this equilibrium the players can recover (learn) the equilibrium with high probability. However, it might be the case that the game has an approximate RCE with a large support from which sampling is easy, while finding (even approximately) the entire joint strategy is hard. In particular, a poly-logarithmic number of samples might not be enough to recover the equilibrium. For example, if one of the players knows a CE of the game, she can sample a pair of actions according to the equilibrium and send the other player his action. However, if the equilibrium she knows has no succinct representation, communicating it might be hard.

The 2-cycle game has a unique exact CE which is the pure Nash equilibrium of the game, and every $1/\text{poly}(N)$-approximate CE of the game is *concentrated* on the pure Nash equilibrium. Thus, by sampling a pair of actions from a $1/\text{poly}(N)$-approximate CE, the players can recover the pure Nash equilibrium with high probability. That is, not only finding a $1/\text{poly}(N)$-approximate CE of the game is hard, but sampling from such an equilibrium is also hard. Since every approximate RCE is an approximate CE, the following lower bound holds also for RCE.

---

[5] In fact, [5] showed that the problem is total for $1/\text{polylog}(N)$ approximation values.

[6] This problem can be naturally extended to sampling from an approximate correlated equilibrium.

▶ **Theorem 2.** *Let $n \geq 3$ be an odd integer, let $N = 2n$, $\Delta \in [0, 2/3)$ and $\varepsilon \leq 1/4N^3(2/3 - \Delta)$. Let μ be an ε-approximate correlated equilibrium of the 2-cycle $N \times N$ game. Then, every randomized communication protocol for sampling from a distribution that is Δ-close in ℓ_1 distance to μ, with error probability at most $\frac{1}{3}$, has communication complexity at least $\Omega(N)$.*

It remains a very interesting open problem to determine the communication complexity of sampling from a *constant*-approximate RCE of two-player games.

1.3 Proof Overview

In the 2-cycle $N \times N$ game, the utility functions are constructed from two subsets of $[n]$ where $n = N/2$. The two subsets have exactly one element in common. Each player is given one of these subsets and computes a directed graph. The two graphs have a common vertex set of size N. The actions of each player are the N vertices. In each graph, every vertex has a unique out-neighbor. Intuitively, each player wants to play the unique out-neighbor (according to his graph) of the vertex played by the other player.

The construction of the utility functions of the 2-cycle game was inspired by ideas of [35] of constructing utility functions from inputs to the fixed-point problem, that is, from continuous functions on a compact convex space. We construct the utility functions in the same way, but from discrete functions, i.e., the unique out-neighbor functions in directed graphs.

To understand how the equilibria of the game look like, we examine the union of the two graphs. An element in the intersection of the subsets creates a directed 2-cycle in the union of the two graphs, with one edge from each graph. Given the two vertices of this 2-cycle, one can recover the index of the element in the intersection of the subsets. The game has a unique (exact) equilibrium which is the two vertices of the 2-cycle (that is, a pure Nash equilibrium). Since it is hard to find the element in the intersection of the subsets, finding an equilibrium of the game is also hard.

The heart of the proof is to show that every $1/\mathrm{poly}(N)$-approximate equilibrium is concentrated on the pure Nash equilibrium. For ease of presentation, we focus on the special case of approximate Nash equilibria. Let (a^*, b^*) be a $1/\mathrm{poly}(N)$-approximate Nash equilibrium of the game. We say that a function $f : [N] \to [0, 1]$ is concentrated on $i \in [N]$ if $f(i) > f(j)$ for every $j \in [N] \setminus \{i\}$. We show that a^*, b^* are concentrated on u^* and v^* respectively, where (u^*, v^*) is the pure Nash equilibrium of the game. Hence given (a^*, b^*), the players can recover the pure Nash equilibrium of the game with no communication. Intuitively, for any vertex v, $a^*(v)$ cannot be large unless one of its neighbors (in the graph of player A) has large probability according to b^*. Similarly, $b^*(v)$ cannot be large unless one of v's neighbors (in the graph of player B) has large probability according to a^*. We use this property to bound a^* and b^* on all the vertices other than u^* and v^* respectively, one by one, moving along alternating edges from the two graphs.

It is interesting to see what happens when the construction of the game is used on two subsets that do not intersect. In this case, the union of the graphs has no 2-cycle and the game has no pure Nash equilibrium. The game has a unique exact Nash equilibrium (a^*, b^*), where a^* is uniform on half of the vertices that correspond to one subset and b^* is uniform on half of the vertices that correspond to the other subset. We note that this is not an equilibrium of the game when the subsets do intersect. For the actual 2-cycle game, constructed from intersecting subsets, we show that every $1/\mathrm{poly}(N)$-approximate equilibrium reveals the pure Nash equilibrium of the game. Thus changing a single bit in the representation of the subsets affects every $1/\mathrm{poly}(N)$-approximate equilibrium of the game. On a more technical note, we

bound a^* and b^* on all vertices other than the ones in the 2-cycle, one by one, starting with a vertex v such that all incoming edges to v are from vertices with small probability according to b^*. Such a vertex does not exist if there is no element in the intersection of the subsets.

1.4 Related Works

We overview previous works related to the computation of correlated equilibria of two-player $N \times N$ games.

Computational complexity

An exact correlated equilibrium can be computed for two-player games in polynomial time by a linear program [20]. Additionally, the decision version of finding correlated equilibria with particular properties have also been considered in literature (for examples see [12, 7]).

Query complexity

Fearnley et al. [10] showed a deterministic query algorithm that finds a $1/2$-approximate Nash equilibrium by making $O(N)$ queries and Fearnley and Savani [11] showed a randomized query algorithm that finds a 0.382-approximate Nash equilibrium by making $O(N \log N)$ queries. For coarse correlated equilibrium, Goldberg and Roth [15] provided a randomized query algorithm that finds a constant approximate coarse correlated equilibrium by making $O(N \log N)$ queries.

Communication complexity

Goldberg and Pastink [14] showed a communication protocol that finds a 0.438-approximate Nash equilibrium by exchanging polylog(N) bits of communication, and Czumaj et al. [9] showed a communication protocol that finds a 0.382-approximate Nash equilibrium with similar communication.

2 Preliminaries

2.1 General Notation

For $n \in \mathbb{N}$, we denote by $[n]$ the set $\{0, 1, \ldots, n-1\}$. For two bit strings $x, y \in \{0,1\}^*$, let xy be the concatenation of x and y. For a bit string $x \in \{0,1\}^n$ and an index $i \in [n]$, x_i is the $i^{th}+1$ bit in x and $\bar{x}$ is the negated bit string, that is $\bar{x}_i$ is the negation of x_i. For a function $\mu : \Omega \to [0,1]$, where Ω is some finite set, and a subset $S \subseteq \Omega$, let

$$\mu(S) = \sum_{z \in S} \mu(z).$$

Define $\mu(\emptyset) = 0$ and $\max_{z \in \emptyset} \mu(z) = 0$. For $u \in \Omega$ we say that μ is *concentrated on* u if

$$\mu(u) > \mu(v) \quad \forall\, v \in \Omega \setminus \{u\}.$$

For a function $\mu : \mathcal{U} \times \mathcal{V} \to [0,1]$, where $\mathcal{U}, \mathcal{V}$ are some finite sets, a subset $S \subseteq \mathcal{U}$ and $v \in \mathcal{V}$, let

$$\mu(S, v) = \sum_{u \in S} \mu(u, v).$$

Similarly, for a subset $S \subseteq \mathcal{V}$ and $u \in \mathcal{U}$ let $\mu(u, S) = \sum_{v \in S} \mu(u, v)$. Define $\mu(\emptyset, v) = \mu(u, \emptyset) = 0$.

2.2 Approximate Correlated Equilibrium

A win-lose, finite game for two players A and B is given by two utility functions $u_A : \mathcal{U} \times \mathcal{V} \to \{0,1\}$ and $u_B : \mathcal{U} \times \mathcal{V} \to \{0,1\}$, where $\mathcal{U}$ and $\mathcal{V}$ are finite sets of actions. We say that the game is an $N \times N$ game, where $N = \max\{|\mathcal{U}|, |\mathcal{V}|\}$. A *mixed strategy* for player A is a distribution over $\mathcal{U}$ and a mixed strategy for player B is a distribution over $\mathcal{V}$. A mixed strategy is called *pure* if it has only one action in its support. A *correlated mixed strategy* is a distribution over $\mathcal{U} \times \mathcal{V}$. A *switching rule* for player A is a mapping from $\mathcal{U}$ to $\mathcal{U}$ and a switching rule for player B is a mapping from $\mathcal{V}$ to $\mathcal{V}$.

▶ **Definition 3** (Approximate Correlated Equilibrium)**.** Let $\varepsilon \in [0,1)$. An ε-approximate correlated equilibrium of a two-player game is a correlated mixed strategy μ such that the following two conditions hold:

1. For all actions $u, u' \in \mathcal{U}$,
$$\sum_{v \in \mathcal{V}} \mu(u,v) \cdot (u_A(u',v) - u_A(u,v)) \leq \varepsilon.$$
2. For all actions $v, v' \in \mathcal{V}$,
$$\sum_{u \in \mathcal{U}} \mu(u,v) \cdot (u_B(u,v') - u_B(u,v)) \leq \varepsilon.$$

▶ **Definition 4** (Approximate Rule Correlated Equilibrium)**.** Let $\varepsilon \in [0,1)$. An ε-approximate rule correlated equilibrium of a two-player game is a correlated mixed strategy μ such that the following two conditions hold:

1. For every switching rule f for player A,
$$\mathbb{E}_{(u,v)\sim\mu}\left[u_A(f(u),v) - u_A(u,v)\right] \leq \varepsilon.$$
2. For every switching rule f for player B,
$$\mathbb{E}_{(u,v)\sim\mu}\left[u_B(u,f(v)) - u_B(u,v)\right] \leq \varepsilon.$$

When the approximation value is zero the two notions above coincide. In general, every approximate rule correlated equilibrium is an approximate correlated equilibrium.

▶ **Proposition 5.** *Fix an N action two-player game and let $\varepsilon \in [0,1)$. Then, every ε-approximate rule correlated equilibrium of the game is an ε-approximate correlated equilibrium of the game. In the other direction, every ε-approximate correlated equilibrium of the game is an $(\varepsilon \cdot N)$-approximate rule correlated equilibrium of the game.*

The communication task of finding an ε-approximate (rule) correlated equilibrium is as follows. Consider a win-lose, finite game for two players A and B, given by two utility functions $u_A : \mathcal{U} \times \mathcal{V} \to \{0,1\}$ and $u_B : \mathcal{U} \times \mathcal{V} \to \{0,1\}$.

Inputs: The actions sets $\mathcal{U}, \mathcal{V}$ and the approximation value ε are known to both players. Player A gets the utility function u_A and player B gets the utility function u_B. The utility functions are given as truth tables of size $|\mathcal{U}| \times |\mathcal{V}|$ each.

At the end of the communication: Both players know the same correlated mixed strategy μ over $\mathcal{U} \times \mathcal{V}$, such that μ is an ε-approximate (rule) correlated equilibrium.

▶ Remark. Note that the communication complexity of a communication protocol for finding an ε-approximate (rule) correlated equilibrium is the total number of bits exchanged between the two players, which might be smaller than the number of bits required to describe the correlated mixed strategy μ to an observer with no prior information.

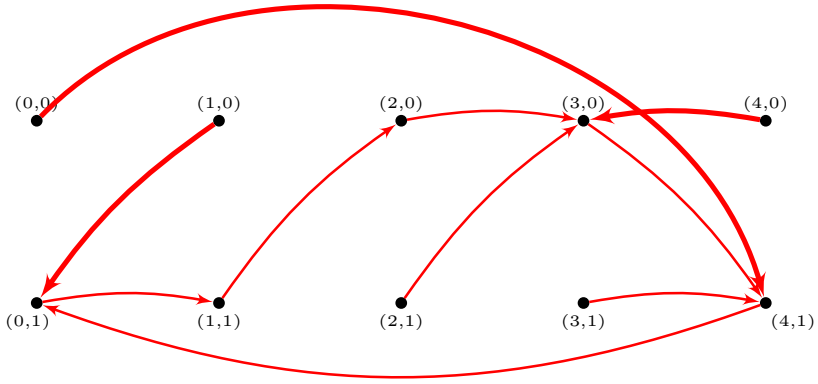


Figure 1 The graph G_A built from the 5 bit string 11001. The thick edges are the edges going back (of the form $((i,0),(i-1,x_{i-1}))$).

3 The 2-Cycle Game

Let $n \geq 3$ be an odd integer. The 2-cycle game is a win-lose, $N \times N$ game, where $N = 2n$. It is constructed from two n-bit strings $x, y \in \{0,1\}^n$ for which there exists exactly one index $i \in [n]$, such that $x_i > y_i$. Throughout the paper, all operations (adding and subtracting) are done modulo n.

The graphs

Given a string $x \in \{0,1\}^n$, player A computes the graph G_A on the set of vertices $V = [n] \times \{0,1\}$ with the following set of directed edges (an edge (u,v) is directed from u into v):

$$\begin{aligned} E_A = &\Big\{((i,1),(i+1,x_{i+1})) : i \in [n]\Big\} \\ &\cup \Big\{((i,0),(i+1,x_{i+1})) : i \in [n], x_i = 0\Big\} \\ &\cup \Big\{((i,0),(i-1,x_{i-1})) : i \in [n], x_i = 1\Big\}. \end{aligned}$$

See an example of such a graph in Figure 1.

Given a string $y \in \{0,1\}^n$, player B computes the graph G_B on the same set of vertices V with the following set of directed edges:

$$E_B = \Big\{((i,z),(i+1,y_{i+1})) : i \in [n], z \in \{0,1\}\Big\}.$$

See an example of such a graph in Figure 2.

The actions and utility functions

The sets of actions are $\mathcal{U} = \mathcal{V} = V$. Intuitively, each player wants to play the unique out-neighbor (according to his graph) of the vertex played by the other player. For example, if player B plays vertex v then player A wants to play the vertex u such that $(v,u) \in E_A$.

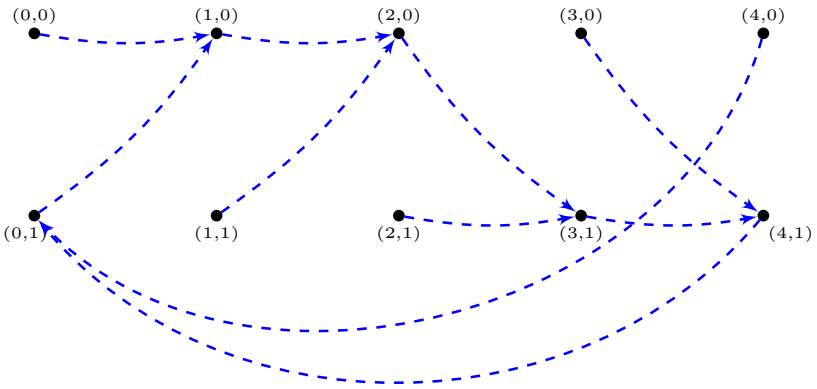


Figure 2 The graph G_B built from the 5 bit string 10011.

Formally, the utility function $u_A : V^2 \to \{0,1\}$ of player A is defined for every pair of actions $(u,v) \in V^2$ as

$$u_A(u,v) = \begin{cases} 1 & \text{if } (v,u) \in E_A \\ 0 & \text{otherwise} \end{cases}.$$

The utility function $u_B : V^2 \to \{0,1\}$ of player B is defined for every pair of actions $(u,v) \in V^2$ as

$$u_B(u,v) = \begin{cases} 1 & \text{if } (u,v) \in E_B \\ 0 & \text{otherwise} \end{cases}.$$

Notations and basic properties

For two vertices $u, v \in V$, (u,v) is a *2-cycle* if $(v,u) \in E_A$ and $(u,v) \in E_B$. For a vertex $u \in V$, define

$$N_A(u) = \{v \in V \ : \ (v,u) \in E_A\}$$
$$N_B(u) = \{v \in V \ : \ (v,u) \in E_B\}.$$

That is, $N_A(u)$ is the set of incoming neighbors to u in E_A, and $N_B(u)$ is the set of incoming neighbors to u in E_B. Let $d_A(u) = |N_A(u)|$ and $d_B(u) = |N_B(u)|$. For a subset $S \subseteq V$, define

$$N_A(S) = \cup_{v \in S} N_A(v)$$
$$N_B(S) = \cup_{v \in S} N_B(v).$$

Edges in E_A of the form $((i,0),(i-1,x_{i-1}))$ for $i \in [n]$ are called *back-edges*. Let x, y be the strings from which the game was constructed. Note that u_A determines x, and u_B determines y. For an index $i \in [n]$ we say that i is *disputed* if $x_i > y_i$. Otherwise, we say that i is *undisputed*. Define i^* to be the unique disputed index. We denote the following key vertices:

$$u^* = (i^* - 1, x_{i^*-1})$$
$$v^* = (i^*, 0) = (i^*, y_{i^*}).$$

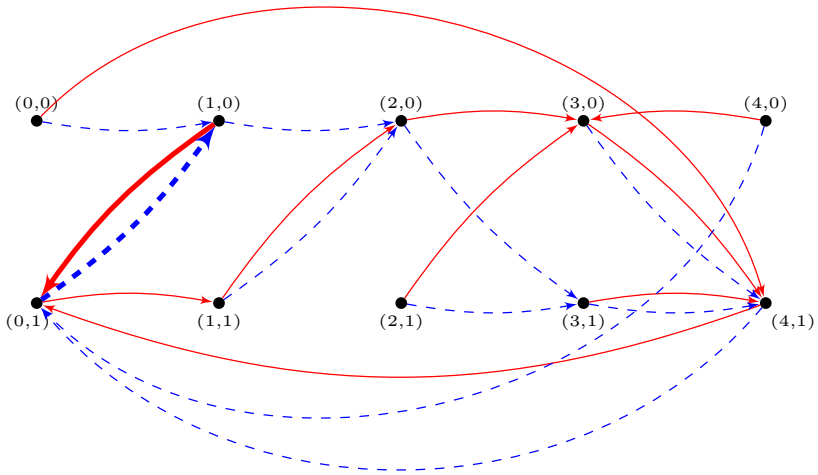


Figure 3 The 2-cycle in the union of the graphs G_A from Figure 1 and G_B from Figure 2.

To simplify notations, for a function f taking inputs from the set V and a vertex $v = (i, z) \in V$, we write $f(i, z)$ instead of $f((i, z))$.

The following are some useful, basic properties of the 2-cycle game.

▶ **Proposition 6** (Out-degree). *For every $v \in V$, there exists exactly one $u \in V$ such that $u_A(u, v) = 1$. Similarly, for every $u \in V$, there exists exactly one $v \in V$ such that $u_B(u, v) = 1$.*

▶ **Proposition 7** (Max in-degree). *For every $v \in V$, it holds that $d_A(v) \leq 3$ and $d_B(v) \leq 2$.*

To understand how the equilibria of the game look like, we will examine the union of the graphs G_A and G_B. The union of the graphs contains a unique 2-cycle, with one edge from G_A and one from G_B. We will see that this 2-cycle corresponds to a pure Nash equilibrium of the game. The 2-cycle in the union of the graphs G_A from Figure 1 and G_B from Figure 2 appears in Figure 3.

▶ **Proposition 8** (A 2-cycle). *Let $(v, u) \in E_A$ be a back-edge. If $v \neq v^*$ then $d_B(v) = 0$. Otherwise, $u = u^*$ and (u^*, v^*) is a 2-cycle.*

Proof. Let $u = (i, z_A) \in V$, for some $z_A \in \{0, 1\}$ and assume there exits $v = (i + 1, z_B) \in N_A(u)$, for some $z_B \in \{0, 1\}$. By the definition of E_A,

$$z_A = x_i, \quad x_{i+1} = 1 \quad \text{and} \quad z_B = 0.$$

If $v \neq v^*$, then $y_{i+1} = 1$ and by the definition of E_B, $d_B(v) = 0$. Otherwise $v = v^*$ and $x_{i^*} > y_{i^*}$. Since $v = v^*$ it holds that $u = u^*$. Since $x_{i^*} > y_{i^*}$ it holds that $y_{i+1} = 0$ and by the definition of E_B, $(u, v) \in E_B$. ◀

3.1 Pure Nash Equilibrium

By Claim 9 below, the 2-cycle game has a unique pure Nash equilibrium. Together with Proposition 8, the pure Nash equilibrium of the game corresponds to the 2-cycle in the union of the two graphs.

▶ **Claim 9.** *The 2-cycle game has exactly one pure Nash equilibrium* (u^*, v^*).

Proof. By Proposition 8, (u^*, v^*) is a 2-cycle. That is, $u_A(u^*, v^*) = 1$ and $u_B(u^*, v^*) = 1$. Since the maximum payoff for either player for any pair of actions is at most 1, it is easy to see that (u^*, v^*) is a pure Nash equilibrium of the game.

Let $u, v \in V$ such that $u \neq u^*$ or $v \neq v^*$. Let a' be the mixed strategy for player A in which she always plays u, and b' be the mixed strategy for player B in which he always plays v. By Proposition 8, either $(v, u) \notin E_A$ or $(u, v) \notin E_B$. By proposition 6, there exist $u', v' \in V$ such that $(v, u') \in E_A$ and $(u, v') \in E_B$. If $(v, u) \notin E_A$ then let a be the mixed strategy for player A in which she always plays u'. We get that

$$\mathbb{E}_{u'' \sim a, v'' \sim b'}[u_A(u'', v'')] = u_A(u', v) = 1 \quad \text{and} \quad \mathbb{E}_{u'' \sim a', v'' \sim b'}[u_A(u'', v'')] = u_A(u, v) = 0.$$

Otherwise $(u, v) \notin E_B$, then let b be the mixed strategy for player B in which he always plays v'. We get that

$$\mathbb{E}_{u'' \sim a', v'' \sim b}[u_B(u'', v'')] = u_B(u, v') = 1 \quad \text{and} \quad \mathbb{E}_{u'' \sim a', v'' \sim b'}[u_B(u'', v'')] = u_B(u, v) = 0.$$

Therefore, (u, v) is not a pure Nash equilibrium. ◀

The following theorem states that finding the pure Nash equilibrium (equivalently, the 2-cycle) of the 2-cycle game is hard. The proof is by a reduction from the following search variant of unique set disjointness: Player A gets a bit string $x \in \{0,1\}^n$ and player B gets a bit string $y \in \{0,1\}^n$. They are promised that there exists exactly one index $i^* \in [n]$ such that $x_{i^*} > y_{i^*}$. Their goal is to find the index i^*. It is well known that the randomized communication complexity of solving this problem with constant error probability is $\Omega(n)$ [3, 23, 31]. This problem is called the *universal monotone relation*. For more details on the universal monotone relation and its connection to unique set disjointness see [25]. Note that a lower bound for finding a pure Nash equilibrium of a different game is already known due to [8].

▶ **Theorem 10.** *Every randomized communication protocol for finding the pure Nash equilibrium of the 2-cycle* $N \times N$ *game, with error probability at most* $\frac{1}{3}$*, has communication complexity at least* $\Omega(N)$.

Proof. Let $x, y \in \{0,1\}^n$ be the inputs to the search variant of unique set disjointness described above. Consider the 2-cycle $N \times N$ game which is constructed from these inputs, given by the utility functions u_A, u_B. Assume towards a contradiction that there exists a communication protocol π for finding the pure Nash equilibrium of the 2-cycle game with error probability at most $1/3$ and communication complexity $o(N)$. The players run π on u_A, u_B and with probability at least $2/3$, at the end of the communication, player A knows u and player B knows v, such that (u, v) is the pure Nash equilibrium of the game. By Claim 9, $u = u^*$ and $v = v^*$. Given u^*, v^* to the players A and B respectively, both players know the index i^*, which is a contradiction. ◀

4 From Approximate Equilibrium to the Pure Nash

In this section we prove Theorem 1. Let $n \geq 3$ be an odd integer, let $N = 2n$ and $\varepsilon \leq 1/4N^3$. Let μ be an ε-approximate correlated equilibrium of the 2-cycle $N \times N$ game, and let x, y be the strings from which the game was constructed. Recall that the pure Nash equilibrium of the game is denoted (u^*, v^*), where $u^* = (i^* - 1, x_{i^*-1})$ and $v^* = (i^*, 0) = (i^*, y_{i^*})$

(see Claim 9). The following theorem implies that μ is concentrated on (u^*, v^*), that is $\mu(u^*, v^*) > \mu(u, v)$ for every $u, v \in V$ such that $u \neq u^*$ or $v \neq v^*$. Therefore given μ, the players know the pure Nash equilibrium with no communication, and Theorem 1 follows from Theorem 10.

▶ **Theorem 11.** *$\mu(u, v) \leq (3N + 1)\varepsilon < 1/N^2$ for every $u, v \in V$ such that $u \neq u^*$ or $v \neq v^*$.*

Next we prove Theorem 11. Let $u, v \in V$ and denote $a(u) = \mu(u, N_A(u))$, $b(v) = \mu(N_B(v), v)$. By Proposition 6, there exists $v' \in V$ such that $u \in N_B(v')$, therefore

$$\mu(u, v) \leq \mu(N_B(v'), v) \leq b(v) + \varepsilon, \tag{1}$$

where the second step follows from the definition of approximate correlated equilibrium (see Definition 3). Similarly it holds that $\mu(u, v) \leq a(u) + \varepsilon$. For every $i \in [n]$, it holds that $N_A(i, \bar{x}_i) = \emptyset$ and $N_B(i, \bar{y}_i) = \emptyset$, therefore $a(i, \bar{x}_i) = b(i, \bar{y}_i) = 0$. We will bound $a(i, x_i)$ for every $i \in [n] \setminus \{i^* - 1\}$ and $b(i, y_i)$ for every $i \in [n] \setminus \{i^*\}$.

▶ **Claim 12.** *For every $i \in [n] \setminus \{i^* - 1\}$, if $(i, y_{i-1}) \in N_A(i, x_i)$ then*

$$a(i, x_i) \leq b(i - 1, y_{i-1}) + 3\varepsilon,$$

otherwise, $a(i, x_i) \leq 3\varepsilon$. For every $i \in [n] \setminus \{i^*\}$,

$$b(i, y_i) \leq a(i - 1, x_{i-1}) + 2\varepsilon.$$

Proof. Let $i \in [n] \setminus \{i^* - 1\}$. By Equation (1), for every $v \in V$ it holds that $\mu((i, x_i), v) \leq b(v) + \varepsilon$. Summing for every $v \in N_A(i, x_i)$ we get that $a(i, x_i) \leq b(N_A(i, x_i)) + 3\varepsilon$, where we bounded the right-hand side using a bound on the maximum in-degree, see Proposition 7. If there exists a back-edge $(v, (i, x_i)) \in E_A$ than by Proposition 8, $d_B(v) = 0$ (that is $N_B(v) = \emptyset$), and $b(v) = 0$. Therefore back-edges do not contribute to the bound on $a(i, x_i)$. It remains to consider edges from $(i - 1, y_{i-1})$ and $(i - 1, \bar{y}_{i-1})$. If $(i, y_{i-1}) \in N_A(i, x_i)$ then

$$a(i, x_i) \leq b(i - 1, y_{i-1}) + b(i - 1, \bar{y}_{i-1}) + 3\varepsilon = b(i - 1, y_{i-1}) + 3\varepsilon,$$

otherwise, $a(i, x_i) \leq b(i - 1, \bar{y}_{i-1}) + 3\varepsilon = 3\varepsilon$. Similarly for every $i \in [n] \setminus \{i^*\}$,

$$\begin{aligned} b(i, y_i) &\leq a(N_B(i, y_i)) + 2\varepsilon \\ &= a(i - 1, x_{i-1}) + a(i - 1, \bar{x}_{i-1}) + 2\varepsilon \\ &= a(i - 1, x_{i-1}) + 2\varepsilon. \end{aligned}$$

◀

Using Claim 12 we can bound $a(i, x_i)$ for every $i \in [n] \setminus \{i^* - 1\}$ and $b(i, y_i)$ for every $i \in [n] \setminus \{i^*\}$ as follows. Let $\delta = 3\varepsilon$. We start with $(i^* + 1, x_{i^*+1})$. Since $x_{i^*} = 1$ and $y_{i^*} = 0$, it holds that $(i^*, y_{i*}) \notin N_A(i^* + 1, x_{i^*+1})$. Therefore by Claim 12,

$$a(i^* + 1, x_{i^*+1}) \leq \delta.$$

Once we bound $a(v)$ (or $b(v)$) for some vertex v, we can apply Claim 12 again to bound the value of b (respectively a) on a neighbor of v. We get that

$$b(i^* + 2, y_{i^*+2}) \leq a(i^* + 1, x_{i^*+1}) + \delta \leq 2\delta,$$

then

$$a(i^* + 3, x_{i^*+3}) \leq b(i^* + 2, y_{i^*+2}) + \delta \leq 3\delta$$

and so on. After we apply Claim 12 n times, since n is odd, we get that $a(i^*, x_{i^*}) \leq n\delta$. We apply Claim 12 $(n-2)$ more times, until get that

$$b(i^* - 2, y_{i^*-2}) \leq n\delta + (n-2)\delta \leq 2n\delta.$$

This concludes the proof as we showed that every $a(i, x_i)$ for $i \in [n] \setminus \{i^* - 1\}$ and every $b(i, y_i)$ for $i \in [n] \setminus \{i^*\}$ is at most $2n\delta = 3N\varepsilon$.

4.1 Sampling from a Correlated Equilibrium

Theorem 2 immediately follows from the fact that the correlated equilibria are concentrated on the pure Nash equilibrium: Let $n \geq 3$ be an odd integer, let $N = 2n$, $\Delta \in [0, 2/3)$ and $\varepsilon \leq 1/4N^3(2/3 - \Delta)$. Let μ be an ε-approximate correlated equilibrium of the 2-cycle $N \times N$ game, and let x, y be the strings from which the game was constructed. Recall that the pure Nash equilibrium of the game is denoted (u^*, v^*), where $u^* = (i^* - 1, x_{i^*-1})$ and $v^* = (i^*, 0) = (i^*, y_{i^*})$ (see Claim 9). By Theorem 11 above, $\mu(u, v) \leq (3N + 1)\varepsilon$ for every $u, v \in V$ such that $u \neq u^*$ or $v \neq v^*$. Thus,

$$\mu(u^*, v^*) \geq 1 - (N^2 - 1)(3N + 1)\varepsilon > 1 - 4N^3\varepsilon \geq \frac{1}{3} + \Delta.$$

If the players can sample from a distribution that is Δ-close in ℓ_1 distance to μ, using $o(N)$ communication bits, then they can find (u^*, v^*) after $O(1)$ attempts with high probability, using $o(N)$ communication bits, in contradiction to Theorem 10.

References

1. Robert Aumann. Subjectivity and correlation in randomized strategies. *Journal of Mathematical Economics*, 1(1):67–96, 1974.
2. Robert Aumann. Correlated equilibrium as an expression of bayesian rationality. *Econometrica*, 55(1):1–18, 1987.
3. László Babai, Peter Frankl, and Janos Simon. Complexity classes in communication complexity theory (preliminary version). In *FOCS*, pages 337–347, 1986.
4. Yakov Babichenko. private communication, 2017.
5. Yakov Babichenko, Siddharth Barman, and Ron Peretz. Empirical distribution of equilibrium play and its testing application. *Math. Oper. Res.*, 42(1):15–29, 2017. doi:10.1287/moor.2016.0794.
6. Yakov Babichenko and Aviad Rubinstein. Communication complexity of approximate Nash equilibria. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 878–889, 2017.
7. Siddharth Barman and Katrina Ligett. Finding any nontrivial coarse correlated equilibrium is hard. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation, EC '15, Portland, OR, USA, June 15-19, 2015*, pages 815–816, 2015. doi:10.1145/2764468.2764497.
8. Vincent Conitzer and Tuomas Sandholm. Communication complexity as a lower bound for learning in games. In *Machine Learning, Proceedings of the Twenty-first International Conference (ICML 2004), Banff, Alberta, Canada, July 4-8, 2004*, 2004. doi:10.1145/1015330.1015351.
9. Artur Czumaj, Argyrios Deligkas, Michail Fasoulakis, John Fearnley, Marcin Jurdzinski, and Rahul Savani. Distributed methods for computing approximate equilibria. In *Web and Internet Economics - 12th International Conference, WINE 2016, Montreal, Canada, December 11-14, 2016, Proceedings*, pages 15–28, 2016. doi:10.1007/978-3-662-54110-4_2.

10 John Fearnley, Martin Gairing, Paul W. Goldberg, and Rahul Savani. Learning equilibria of games via payoff queries. *Journal of Machine Learning Research*, 16:1305–1344, 2015. URL: http://dl.acm.org/citation.cfm?id=2886792.

11 John Fearnley and Rahul Savani. Finding approximate Nash equilibria of bimatrix games via payoff queries. *ACM Trans. Economics and Comput.*, 4(4):25:1–25:19, 2016. doi:10.1145/2956579.

12 Itzhak Gilboa and Eitan Zemel. Nash and correlated equilibria: Some complexity considerations. *Games and Economic Behavior*, 1(1):80–93, 1989. doi:10.1016/0899-8256(89)90006-7.

13 Paul Goldberg. *Surveys in Combinatorics 2011*. London Mathematical Society Lecture Note Series, 2011.

14 Paul W. Goldberg and Arnoud Pastink. On the communication complexity of approximate Nash equilibria. *Games and Economic Behavior*, 85:19–31, 2014. doi:10.1016/j.geb.2014.01.009.

15 Paul W. Goldberg and Aaron Roth. Bounds for the query complexity of approximate equilibria. In *ACM Conference on Economics and Computation, EC '14, Stanford , CA, USA, June 8-12, 2014*, pages 639–656, 2014. doi:10.1145/2600057.2602845.

16 Sergiu Hart and Yishay Mansour. How long to equilibrium? the communication complexity of uncoupled equilibrium procedures. *Games and Economic Behavior*, 69(1):107–126, 2010. doi:10.1016/j.geb.2007.12.002.

17 Sergiu Hart and Andreu Mas-Colell. Uncoupled dynamics do not lead to Nash equilibrium. *American Economic Review*, 93(5):1830–1836, 2003.

18 Sergiu Hart and Andreu Mas-Colell. Stochastic uncoupled dynamics and Nash equilibrium. *Games and Economic Behavior*, 57(2):286–303, 2006.

19 Sergiu Hart and Andreu Mas-Colell. *Simple Adaptive Strategies:From Regret-Matching to Uncoupled Dynamics*. World Scientific Publishing Co. Pte. Ltd., 2013.

20 Sergiu Hart and David Schmeidler. Existence of correlated equilibria. *Math. Oper. Res.*, 14(1):18–25, 1989. doi:10.1287/moor.14.1.18.

21 Albert Xin Jiang and Kevin Leyton-Brown. Polynomial-time computation of exact correlated equilibrium in compact games. *Games and Economic Behavior*, 91:347–359, 2015. doi:10.1016/j.geb.2013.02.002.

22 Ehud Kalai and Ehud Lehrer. Rational learning leads to nash equilibrium. *Econometrica*, 61(5):1019–45, 1993.

23 Bala Kalyanasundaram and Georg Schnitger. The probabilistic communication complexity of set intersection. *SIAM J. Discrete Math.*, 5(4):545–557, 1992.

24 Young Kun Ko and Ariel Schvartzman. Bounds for the communication complexity of two-player approximate correlated equilibria. *Electronic Colloquium on Computational Complexity (ECCC)*, 24:71, 2017. URL: https://eccc.weizmann.ac.il/report/2017/071.

25 Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, New York, NY, USA, 1997.

26 Troy Lee and Adi Shraibman. Lower bounds in communication complexity. *Foundations and Trends in Theoretical Computer Science*, 3(4):263–398, 2009.

27 Moni Naor. private communication, 2017.

28 J.F. Nash. Non-cooperative games. *Annals of Mathematics*, 54(2):286–295, 1951.

29 Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V Vazirani. *Algorithmic Game Theory*. Cambridge University Press, New York, NY, USA, 2007.

30 Christos H. Papadimitriou and Tim Roughgarden. Computing correlated equilibria in multi-player games. *J. ACM*, 55(3):14:1–14:29, 2008. doi:10.1145/1379759.1379762.

31 Alexander A. Razborov. On the distributional complexity of disjointness. *Theor. Comput. Sci.*, 106(2):385–390, 1992.

32 Tim Roughgarden. Computing equilibria: a computational complexity perspective. *Economic Theory*, 42(1):193–236, 2010. doi:10.1007/s00199-009-0448-y.

33 Tim Roughgarden. Communication complexity (for algorithm designers). *Foundations and Trends in Theoretical Computer Science*, 11(3-4):217–404, 2016. doi:10.1561/0400000076.

34 Tim Roughgarden. *Twenty Lectures on Algorithmic Game Theory*. Cambridge University Press, 2016.

35 Tim Roughgarden and Omri Weinstein. On the communication complexity of approximate fixed points. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 229–238, 2016. doi:10.1109/FOCS.2016.32.

36 Andrew Chi-Chih Yao. Some complexity questions related to distributive computing (preliminary report). In *Proceedings of the 11h Annual ACM Symposium on Theory of Computing, April 30 - May 2, 1979, Atlanta, Georgia, USA*, pages 209–213, 1979.

37 H. Peyton Young. H. peyton young, , strategic learning and its limits (2004) oxford univ. press 165 pages. *Games and Economic Behavior*, 63(1):417–420, 2004.

A Trivial Approximate Equilibria of The 2-Cycle Game

In this section, we provide trivial approximate correlated equilibrium of the 2-cycle game from which it is not possible to recover the disputed index.

Let us suppose that for all $i \in \left[\frac{n}{2}+3\right]$, we have $x_i = y_i = 0$.

We define a joint distribution μ as follows

$$\mu((i,z_A),(j,z_B)) = \begin{cases} \frac{16\alpha}{n^2} \text{ if } z_A, z_B = 0 \text{ and } \frac{n}{4}+4 \le i,j \le \frac{n}{2}+2, \\ \frac{16\alpha}{n^2} \text{ if } z_A, z_B = 0, \frac{n}{4}+2 \le j \le \frac{n}{2}+2 \text{ and } i = \frac{n}{4}+3, \\ \frac{16\alpha}{n^2} \text{ if } z_A, z_B = 0, \frac{n}{4}+2 \le i \le \frac{n}{2}+2 \text{ and } j = \frac{n}{4}+3, \\ \frac{16\alpha}{n^2} - \frac{64\alpha\cdot(n/4-i+3)}{n^3} \text{ if } z_A, z_B = 0, 2 \le i,j \le \frac{n}{4}+2 \text{ and } i-j=1, \\ \frac{16\alpha}{n^2} - \frac{64\alpha\cdot(n/4-j+3)}{n^3} \text{ if } z_A, z_B = 0, 2 \le i,j \le \frac{n}{4}+2 \text{ and } j-i=1, \\ 0 \text{ otherwise,} \end{cases}$$

where α is some normalizing constant less than 2 such that $\sum_{(u,v)\in V^2} \mu(u,v) = 1$.

Let $\varepsilon = 64\alpha/n^3$. For every action $u = (i, z_A)$ of Alice such that $z_A \neq 0$, we have that $\mu(u,v) = 0$ for all $v \in V$. Similarly for every action $v = (j, z_B)$ of Bob such that $z_B \neq 0$, we have that $\mu(u,v) = 0$ for all $u \in V$. Also, for every action $u = (i, z_A)$ of Alice such that $i \in \{n/2+3, \ldots, n\} \cup \{1\}$, we have that $\mu(u,v) = 0$ for all $v \in V$. And, similarly for every action $v = (j, z_B)$ of Bob such that $j \in \{n/2+3, \ldots, n\} \cup \{1\}$, we have that $\mu(u,v) = 0$ for all $u \in V$. Since μ is symmetric[7], it follows that in order to show that μ is an ε-approximate correlated equilibrium we only need to consider a vertex $u = (i, 0)$ when $i \in \left[\frac{n}{2}+2\right]$.

First, we consider when $i \le \frac{n}{4}+2$. Let $u' \in V$. We have

$$\begin{aligned} \sum_{v\in V} \mu(u,v)\cdot(u_A(u',v) - u_A(u,v)) &= \mu(u, N_A(u')) - \mu(u, N_A(u)) \\ &= \mu(u, N_A(u')) - \frac{16\alpha}{n^2} + \frac{64\alpha\cdot(n/4-i+3)}{n^3}. \end{aligned}$$

[7] i.e., $\mu(u,v) = \mu(v,u)$ for all $u, v \in V$.

Now if $v = (j, z_B) \in N_A(u')$ and $|j - i| \neq 1$ then, we have $\mu(u, v) = 0$. Thus, we assume $j - i = 1$, as we suppose $u \neq u'$. Then, we have

$$\begin{aligned}\mu(u, N_A(u')) &\leq \frac{16\alpha}{n^2} - \frac{64\alpha \cdot (n/4 - i - 1 + 3)}{n^3} \\ &= \frac{16\alpha}{n^2} - \frac{64\alpha \cdot (n/4 - i + 3)}{n^3} + \frac{64\alpha}{n^3}.\end{aligned}$$

This implies,

$$\sum_{v \in V} \mu(u, v) \cdot (u_A(u', v) - u_A(u, v)) \leq \frac{64\alpha}{n^3} = \varepsilon.$$

Next, we consider when $\frac{n}{4} + 4 \leq i \leq \frac{n}{2} + 2$. Let $u' \in V$. We have

$$\begin{aligned}\sum_{v \in V} \mu(u, v) \cdot (u_A(u', v) - u_A(u, v)) &= \mu(u, N_A(u')) - \mu(u, N_A(u)) \\ &= \mu(u, N_A(u')) - \frac{16\alpha}{n^2}.\end{aligned}$$

Now if $v = (j, z_B) \in N_A(u')$ and $j \geq \frac{n}{2} + 3$ then, we have $\mu(u, v) = 0$. Also if $j \leq \frac{n}{4} + 2$ then, we have $\mu(u, v) = 0$. Thus, we assume $j \in [n/4 + 3, n/4 + 2]$ and $\beta = 0$. Then, we have

$$\mu(u, N_A(u')) \leq \frac{16\alpha}{n^2}.$$

This implies,

$$\sum_{v \in V} \mu(u, v) \cdot (u_A(u', v) - u_A(u, v)) \leq 0.$$

Finally, we consider when $i = \frac{n}{4} + 3$. Let $u' = (i', z'_A) \in V$. We have

$$\sum_{v \in V} \mu(u, v) \cdot (u_A(u', v) - u_A(u, v)) = \mu(u, N_A(u')) - \frac{16\alpha}{n^2} + \frac{64\alpha}{n^3}.$$

Now if $v = (j, z_B) \in N_A(u')$ and $j \geq \frac{n}{2} + 3$ then, we have $\mu(u, v) = 0$. Also if $j \leq \frac{n}{4} + 2$ and $|j - i| \neq 1$ then, we have $\mu(u, v) = 0$. Since $u \neq u'$ we have that $j \in [n/4 + 3, n/4 + 2]$ and $\beta = 0$. Then we have

$$\mu(u, N_A(u')) \leq \frac{16\alpha}{n^2}.$$

This implies,

$$\sum_{v \in V} \mu(u, v) \cdot (u_A(u', v) - u_A(u, v)) \leq \frac{64\alpha}{n^3} = \varepsilon.$$

Thus, μ is an ε-approximate correlated equilibrium and an $(\varepsilon \cdot N)$-approximate rule correlated equilibrium.

On Minrank and the Lovász Theta Function

Ishay Haviv
School of Computer Science, The Academic College of Tel Aviv-Yaffo, Tel Aviv 61083, Israel

Abstract

Two classical upper bounds on the Shannon capacity of graphs are the ϑ-function due to Lovász and the minrank parameter due to Haemers. We provide several explicit constructions of n-vertex graphs with a constant ϑ-function and minrank at least n^{δ} for a constant $\delta > 0$ (over various prime order fields). This implies a limitation on the ϑ-function-based algorithmic approach to approximating the minrank parameter of graphs. The proofs involve linear spaces of multivariate polynomials and the method of higher incidence matrices.

2012 ACM Subject Classification Mathematics of computing → Information theory

Keywords and phrases Minrank, Theta Function, Shannon capacity, Multivariate polynomials, Higher incidence matrices

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.13

1 Introduction

For a graph G on the vertex set V, let G^k denote the graph on the vertex set V^k in which two distinct vertices $(u_1, \ldots, u_k)$ and $(v_1, \ldots, v_k)$ are adjacent if for every $1 \leq i \leq k$ it holds that u_i and v_i are either equal or adjacent in G. The *Shannon capacity* of G, introduced by Shannon in 1956 [43], is defined as the limit $c(G) = \lim_{k\to\infty} \sqrt[k]{\alpha(G^k)}$, where $\alpha(G^k)$ stands for the independence number of G^k. The study of the graph parameter $c(G)$ is motivated by an application in information theory, as it measures the effective alphabet size in a communication over a noisy channel represented by G. However, computing the Shannon capacity of a graph is a notoriously difficult task. Its exact value is not known even for small graphs, such as the cycle on 7 vertices, and from a computational perspective, it is not known if the problem of deciding whether the Shannon capacity of a given graph exceeds a given value is decidable.

The difficulty in computing the Shannon capacity of graphs motivates studying upper and lower bounds on $c(G)$. It is known that $c(G)$ is sandwiched between the independence number $\alpha(G)$ of G and its clique cover number $\chi(\overline{G})$. In 1979, Lovász [34] introduced the ϑ-function of graphs defined as follows: For a graph G on the vertex set V, $\vartheta(G)$ is the minimum of $\max_{i\in V} \frac{1}{\langle x_i, y\rangle^2}$, taken over all choices of unit vectors y and $(x_i)_{i\in V}$ such that x_i and x_j are orthogonal whenever i and j are distinct non-adjacent vertices in G (see [32] for several equivalent definitions). It was shown in [34] that $c(G) \leq \vartheta(G)$ for every graph G, and this was used to prove that the Shannon capacity of the cycle on 5 vertices is equal to $\sqrt{5}$. The ϑ-function of graphs can be computed in polynomial running time at an arbitrary precision using semi-definite programming [23] and it has found interesting combinatorial and algorithmic applications over the years (see, e.g., [18, 4, 38]).

Another upper bound on the Shannon capacity of graphs is the minrank parameter introduced by Haemers [24, 25]. For a graph G on the vertex set $V = \{1, \ldots, n\}$, the minrank of G over a field $\mathbb{F}$, denoted $\mathrm{minrk}_{\mathbb{F}}(G)$, is the minimum of $\mathrm{rank}_{\mathbb{F}}(M)$ over all matrices $M \in \mathbb{F}^{n\times n}$ satisfying $M_{i,i} \neq 0$ for every $i \in V$, and $M_{i,j} = 0$ whenever i and j are distinct non-adjacent vertices in G. For the field $\mathbb{F}_p$ of prime order p we use the notation $\mathrm{minrk}_p(G)$. For most graphs the minrank parameter is larger than the ϑ-function [14, 26, 22], yet it was

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018).
Editors: Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer; Article No. 13; pp. 13:1–13:15

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

shown in [24] that there are graphs for which the minrank bound on the Shannon capacity is tighter. In recent years, the minrank parameter has attracted an intensive research motivated by its relations to various topics in information theory, e.g., (linear) index coding [9, 7, 35, 10], network coding [1, 17], distributed storage [37, 29], and wireless communication [36, 28], and in theoretical computer science, e.g., Valiant's approach to lower bounds in circuit complexity [44, 42, 22], communication complexity [40], and randomized computation [27].

The computational problem of deciding whether the minrank of a given graph is at most 3 is known to be NP-complete over any fixed finite field [39]. Moreover, assuming a certain variant of the unique games conjecture, it is NP-hard to approximate the minrank of a given graph to within any constant [33] (and even to within certain super-constant factors, as follows from [33] combined with [15]). On the algorithmic side, relations between the minrank parameter and the tractable ϑ-function can be beneficial to efficient approximation algorithms for minrank. This approach was taken in [13] where it was proved that every graph G with $\mathrm{minrk}_2(G) = k$ satisfies $\vartheta(G) \leq 2^{k/2} + 1 - 2^{1-k/2}$. This bound was used to obtain an efficient algorithm that given an n-vertex graph G with $\mathrm{minrk}_2(G) = k$, where k is a constant, finds a clique cover of G of size $O(n^{\alpha(k)})$ for some $\alpha(k) < 1$ (e.g., $\alpha(3) \approx 0.2574$). Note that such a clique cover of G in particular yields a matrix confirming the same bound on the minrank.

The algorithm of [13] for minrank employs the semi-definite programming technique used in the algorithm of Karger, Motwani, and Sudan for graph coloring [31]. The analysis of the latter shows that every n-vertex graph G with a constant $\vartheta(G)$ has a clique cover of size $O(n^{\alpha})$ for a constant $\alpha < 1$. This is known to be tight in the sense that there are n-vertex graphs G with a constant $\vartheta(G)$ and yet a clique cover number $n^{\Omega(1)}$ [31, 12] (see also [19]). However, the minrank of a graph might in general be much smaller than its clique cover number (even exponentially; see [25]). It is natural to ask, then, whether a constant $\vartheta(G)$ guarantees a stronger bound of $n^{o(1)}$ on $\mathrm{minrk}_2(G)$. In the current work we rule out this possibility in a general sense, as stated below.

▶ **Theorem 1.** *For every prime p there exist $c = c(p)$ and $\delta = \delta(p) > 0$ such that for infinitely many integers n there exists an n-vertex graph G such that $\vartheta(G) \leq c$ and $\mathrm{minrk}_p(G) \geq n^{\delta}$.*

Note that for the special case of $p = 2$ we obtain an n-vertex graph G with $\vartheta(G) \leq 16$ and $\mathrm{minrk}_2(G) \geq n^{0.1499}$. This implies a limitation on the ϑ-function-based algorithmic approach of [13] to minrank over $\mathbb{F}_2$.

We also obtain the following result in which the prime p is not a constant.

▶ **Theorem 2.** *There exist c and $\delta > 0$ such that for infinitely many integers n there exists an n-vertex graph G such that $\vartheta(G) \leq c$ and $\mathrm{minrk}_p(G) \geq n^{\delta}$ for some prime $p = \Theta(\log n)$.*

In our final construction, the bound on the minrank holds over any field of a sufficiently large prime order. However, the bound on the ϑ-function is relaxed to a bound on the vector chromatic number χ_v of the graph's complement (see Definition 6).

▶ **Theorem 3.** *There exists a constant $\delta > 0$ such that for infinitely many integers n there exists an n-vertex graph G such that $\chi_v(\overline{G}) \leq 3$ and $\mathrm{minrk}_p(G) \geq n^{\delta}$ for any prime $p \geq \Omega(\log n)$.*

All the aforementioned constructions are explicit and belong to the family of generalized Kneser graphs (see Definition 8). Our technical contribution lies in presenting two general methods for proving bounds on the minrank parameter, employing the tools of linear spaces of multivariate polynomials and higher incidence matrices (see, e.g., [6, Chapters 5 and 7]).

We demonstrate the usefulness of these tools in studying the minrank of additional graph families (see Sections 3.2 and 3.3) and expect our techniques to have further applications in the future.

1.1 Techniques and Related Work

As mentioned above, the constructions given in Theorems 1, 2, and 3 are all from the family of generalized Kneser graphs. In these graphs the vertices are all subsets of a given size of some universe, and two distinct vertices are adjacent if their intersection size lies in a certain specified set of sizes. We are particularly interested in those graphs with only one intersection size in the specified set, as the ϑ-function of their complement is easily bounded (see Lemma 9).

The independence numbers of generalized Kneser graphs correspond to well-studied combinatorial questions on the size of uniform set systems with forbidden intersection sizes (see, e.g., [20]). Tools from linear algebra are often used in proving upper bounds in such scenarios. This includes the celebrated works of Ray-Chaudhuri and Wilson [41] and Frankl and Wilson [21], who obtained their bounds using the method of higher incidence matrices (more specifically, inclusion matrices; see [6, Chapter 7]). Alon, Babai, and Suzuki [3] provided alternative proofs and generalizations using a different approach operating on linear spaces of multivariate polynomials (see [6, Chapter 5]). These results have found numerous applications in combinatorics and in theoretical computer science, e.g., explicit constructions in Euclidean Ramsey theory [5], counterexamples to Borsuk's conjecture [30], and integrality gap constructions for approximating graph parameters such as the chromatic number [31], the independence number [18, 4], and the vertex cover number [12].

While the above results provide strong upper bounds on the independence numbers of certain generalized Kneser graphs, they do not imply any meaningful bounds on their minrank. Nevertheless, we show in this work that both the tools of higher incidence matrices and multivariate polynomials can be used to obtain upper bounds on the minrank parameter as well. We demonstrate these techniques and apply them to several graph families (most, but not all, of which are of the Kneser type). To obtain the lower bounds on the minrank in Theorems 1, 2, and 3, we apply a known relation between the minrank of a graph and the minrank of its complement (see Lemma 5).

We note that Alon used in [2] multivariate polynomials to obtain an upper bound, closely related to minrank, on the Shannon capacity of graphs. In addition, Lubetzky and Stav used inclusion matrices in [35] to prove that for every prime p, an n-vertex graph can have a multiplicative gap of $n^{0.5-o(1)}$, in either direction, between the ϑ-function and the minrank over $\mathbb{F}_p$. (Note that the bound on ϑ in these constructions is of $\sqrt{n}$.) It will be interesting to figure out if our construction in Theorem 1 combined with the randomized graph product technique of [8, 18] can be used to improve on this multiplicative gap.

Outline.

In Section 2 we gather a few needed definitions and lemmas. In Sections 3 and 4 we prove upper bounds on the minrank of several graph families using, respectively, linear spaces of multivariate polynomials and inclusion matrices. Finally, in Section 5, we prove Theorems 1, 2, and 3.

2 Preliminaries

Unless otherwise specified, a graph will refer to a simple undirected graph. We use the notation $[d] = \{1, 2, \ldots, d\}$.

2.1 Minrank

The minrank of a graph over a field $\mathbb{F}$ is defined as follows.

▶ **Definition 4.** Let $G = (V, E)$ be a directed graph on the vertex set $V = \{1, \ldots, n\}$ and let $\mathbb{F}$ be a field. We say that an n by n matrix M over $\mathbb{F}$ *represents* G if $M_{i,i} \neq 0$ for every $i \in V$, and $M_{i,j} = 0$ for every distinct $i, j \in V$ such that $(i, j) \notin E$. The *minrank* of G over $\mathbb{F}$ is defined as

$$\text{minrk}_{\mathbb{F}}(G) = \min\{\text{rank}_{\mathbb{F}}(M) \mid M \text{ represents } G \text{ over } \mathbb{F}\}.$$

The above definition is naturally extended to undirected graphs by replacing every undirected edge with two oppositely directed edges. Note that for a prime p we write $\text{rank}_p(M) = \text{rank}_{\mathbb{F}_p}(M)$ and $\text{minrk}_p(G) = \text{minrk}_{\mathbb{F}_p}(G)$.

We need the following lemma that relates the minrank of a graph to the minrank of its complement. For a proof see, e.g., [39, Remark 2.2], [35, Claim 2.5].

▶ **Lemma 5.** *For every field $\mathbb{F}$ and an n-vertex graph G,* $\text{minrk}_{\mathbb{F}}(G) \cdot \text{minrk}_{\mathbb{F}}(\overline{G}) \geq n$.

2.2 Vector Chromatic Number

Consider the following two relaxations of the chromatic number of a graph, due to Karger, Motwani, and Sudan [31].

▶ **Definition 6.** For a graph $G = (V, E)$ the *vector chromatic number* of G, denoted $\chi_v(G)$, is the minimal real value of $\kappa > 1$ such that there exists an assignment of a unit vector w_i to each vertex $i \in V$ satisfying the inequality $\langle w_i, w_j \rangle \leq -\frac{1}{\kappa - 1}$ whenever i and j are adjacent in G.

▶ **Definition 7.** For a graph $G = (V, E)$ the *strict vector chromatic number* of G, denoted $\chi_v^{(s)}(G)$, is the minimal real value of $\kappa > 1$ such that there exists an assignment of a unit vector w_i to each vertex $i \in V$ satisfying the equality $\langle w_i, w_j \rangle = -\frac{1}{\kappa - 1}$ whenever i and j are adjacent in G.

It is well known and easy to verify that for every graph G, $\chi_v(G) \leq \chi_v^{(s)}(G) \leq \chi(G)$. The Lovász ϑ-function, introduced in [34], is known to satisfy $\vartheta(G) = \chi_v^{(s)}(\overline{G})$ for every graph G [31] (see Section 1 for its original definition).

2.3 Generalized Kneser Graphs

Consider the family of generalized Kneser graphs defined below. In these graphs the vertices are subsets of some universe and the existence of an edge connecting two sets is decided according to their intersection size.

▶ **Definition 8.** For integers $s \leq d$ and a set $T \subseteq \{0, 1, \ldots, s - 1\}$, the graph $K(d, s, T)$ is defined as follows: the vertices are all possible s-subsets of a universe $[d]$ (i.e., subsets of $[d]$ of size s), and two distinct sets A, B are adjacent if $|A \cap B| \in T$.

The following lemma provides a bound on the strict and non-strict vector chromatic numbers of certain generalized Kneser graphs. Its proof can be found in [31, Section 9], and we include it here for completeness.

▶ **Lemma 9** ([31]). *Let $t < s < d$ be integers satisfying $s^2 > dt$.*

1. *If $T = \{0, 1, \ldots, t\}$ then $\chi_v(K(d, s, T)) \leq \frac{d(s-t)}{s^2-dt}$.*
2. *If $T = \{t\}$ then $\chi_v^{(s)}(K(d, s, T)) \leq \frac{d(s-t)}{s^2-dt}$.*

Proof. Associate every vertex A of $K(d, s, T)$, representing an s-subset of $[d]$, with the vector $u_A \in \mathbb{R}^d$ defined by

$$(u_A)_i = z \;\text{ if }\; i \in A \;\text{ and }\; (u_A)_i = -1 \;\text{ if }\; i \notin A, \;\text{ for every }\; i \in [d],$$

where z is a positive real number to be determined. Notice that $\|u_A\|^2 = s \cdot z^2 + d - s$. Denote by $w_A \in \mathbb{R}^d$ the unit vector defined by $w_A = u_A/\|u_A\|$.

We start with Item 1. Let $T = \{0, 1, \ldots, t\}$. Every two adjacent vertices A and B in $K(d, s, T)$ satisfy $|A \cap B| \leq t$, hence $|A \triangle B| \geq 2(s - t)$ and $|\overline{A \cup B}| \leq d - 2s + t$. It follows that

$$\begin{aligned}
\langle w_A, w_B \rangle &= \frac{1}{s \cdot z^2 + d - s} \cdot \langle u_A, u_B \rangle \\
&= \frac{1}{s \cdot z^2 + d - s} \cdot \Big(|A \cap B| \cdot z^2 - |A \triangle B| \cdot z + |\overline{A \cup B}| \Big) \\
&\leq \frac{t \cdot z^2 - 2(s-t) \cdot z + d - 2s + t}{s \cdot z^2 + d - s}.
\end{aligned}$$

A straightforward calculation shows that the minimum of the above expression is attained at $z = \frac{d}{s} - 1$ and is equal to $-\frac{1}{\kappa - 1}$ for $\kappa = \frac{d(s-t)}{s^2-dt} > 1$. This completes the proof of Item 1. The proof of Item 2 is essentially identical. For adjacent vertices A and B in $K(d, s, T)$ where $T = \{t\}$ we have $|A \cap B| = t$, hence the above upper bound on $\langle w_A, w_B \rangle$ is tight, as needed for the strict vector chromatic number. ◀

2.4 Linear Algebra Fact

▶ **Fact 10.** *Let p be a prime and let M be an integer matrix. Then, the matrix M' defined by $M' = M \pmod p$ satisfies $\text{rank}_p(M') \leq \text{rank}_{\mathbb{R}}(M)$.*

Proof. It suffices to show that if some rows $v_1, \ldots, v_k$ of M are linearly dependent over $\mathbb{R}$ then, considered modulo p, they are also linearly dependent over $\mathbb{F}_p$. To see this, assume that there exist $a_1, \ldots, a_k \in \mathbb{R}$, at least one of which is nonzero, for which $\sum_{i=1}^k a_i v_i = 0$. Since the v_i's are integer vectors it can be assumed that $a_1, \ldots, a_k \in \mathbb{Z}$ and that $\gcd(a_1, \ldots, a_k) = 1$. This implies that they are not all zeros modulo p. Therefore, the same coefficients, considered modulo p, provide a non-trivial combination of the corresponding rows of M' with sum zero, and we are done. ◀

3 Upper Bounds on Minrank via Multivariate Polynomials

In this section we prove upper bounds on the minrank parameter of graphs using linear spaces of multivariate polynomials. We first introduce the notion of functional bi-representations of graphs.

▶ **Definition 11.** Let $G = (V, E)$ be a directed graph and let $\mathbb{F}$ be a field. A *functional bi-representation* of G over $\mathbb{F}$ of dimension R is an assignment of two functions $g_i, h_i : V \to \mathbb{F}$ to each $i \in [R]$ such that the function $f : V \times V \to \mathbb{F}$ defined by

$$f(u, v) = \sum_{i=1}^{R} g_i(u) h_i(v) \tag{1}$$

satisfies

1. $f(v, v) \neq 0$ for every $v \in V$, and
2. $f(u, v) = 0$ for every distinct $u, v \in V$ such that $(u, v) \notin E$.

Note that the definition is naturally extended to undirected graphs.

Functional bi-representations can be used to provide an alternative definition for the minrank parameter, as stated below.

▶ **Lemma 12.** *For every (directed) graph G and a field $\mathbb{F}$, $\mathrm{minrk}_{\mathbb{F}}(G)$ is the smallest integer R for which there exists a functional bi-representation of G over $\mathbb{F}$ of dimension R.*

Observe that Lemma 12 follows directly from Definition 4 and the linear algebra fact that the rank of a matrix $M \in \mathbb{F}^{N \times N}$ is the smallest R for which $M = A \cdot B$ for two matrices $A \in \mathbb{F}^{N \times R}$ and $B \in \mathbb{F}^{R \times N}$, where the functions g_i and h_i in Definition 11 correspond to the columns and the rows of such A and B respectively. A similar definition of the minrank parameter was previously used by Peeters in [39] (see also [13]), where the role of the functions in Definition 11 was taken by vectors.

3.1 Generalized Kneser Graphs

We prove now upper bounds on the minrank of generalized Kneser graphs $K(d, s, T)$ (recall Definition 8). The proofs borrow ideas from [3] and [2]. We start with an upper bound on the minrank of $K(d, s, T)$ over $\mathbb{F}_p$ for all sufficiently large primes p. Note that a slightly improved bound is given in Section 4 (see Proposition 20).

▶ **Proposition 13.** *For every integers $t \leq s \leq d$, a set $T \subseteq \{0, 1, \ldots, s-1\}$ of size $|T| = t$, and a prime $p > s$,*

$$\mathrm{minrk}_p(K(d, s, T)) \leq \sum_{i=0}^{s-t} \binom{d}{i}.$$

Proof. Let $p > s$ be a prime, and let $f : \{0,1\}^d \times \{0,1\}^d \to \mathbb{F}_p$ be the function defined by

$$f(x, y) = \prod_{j \in \{0,1,\ldots,s-1\} \setminus T} \Big(\sum_{i=1}^{d} x_i y_i - j \Big) \pmod p$$

for every $x, y \in \{0,1\}^d$. Expanding f as a linear combination of monomials, the relation $z^2 = z$ for $z \in \{0, 1\}$ implies that one can reduce to 1 the exponent of each variable occurring in a monomial. It follows that f can be represented as a multilinear polynomial in the $2d$ variables of x and y. By combining terms involving the same monomial in the variables of x, one can write f as in (1) for an integer R and functions $g_i, h_i : \{0,1\}^d \to \mathbb{F}_p$ such that the g_i's are distinct multilinear monomials of total degree at most $s - t$ in d variables. It follows that $R \leq \sum_{i=0}^{s-t} \binom{d}{i}$.

Now, denote by V the vertex set of the graph $K(d, s, T)$ and identify each vertex $X \in V$ with an indicator vector $c_X \in \{0,1\}^d$ in the natural way. We observe that the functions g_i

and h_i restricted to V form a functional bi-representation of $K(d,s,T)$ over $\mathbb{F}_p$. Indeed, for every two vertices $A, B \in V$ we have $f(c_A, c_B) = \prod_{j\in\{0,1,\ldots,s-1\}\setminus T}(|A\cap B| - j) \pmod p$. If A and B are distinct and non-adjacent in $K(d,s,T)$ then $|A\cap B| \in \{0,1,\ldots,s-1\}\setminus T$, and thus $f(c_A,c_B) = 0$. On the other hand, every vertex A satisfies $|A| = s$ and thus, using the assumption $p > s$, $f(c_A,c_A) \neq 0$. By Lemma 12 it follows that $\text{minrk}_p(K(d,s,T)) \leq R$, and we are done. ◀

We next consider a special case of the generalized Kneser graphs corresponding to one intersection size.

▶ **Proposition 14.** *For every prime p and integer $d \geq 2p-1$, $\text{minrk}_p(K(d,2p-1,\{p-1\})) \leq \sum_{i=0}^{p-1}\binom{d}{i}$.*

Proof. For a prime p and an integer $d \geq 2p-1$, consider the graph $G = K(d,2p-1,\{p-1\})$. Let $f : \{0,1\}^d \times \{0,1\}^d \to \mathbb{F}_p$ be the function defined by

$$f(x,y) = \prod_{j=0}^{p-2}\Big(\sum_{i=1}^{d} x_i y_i - j\Big) \pmod p$$

for every $x, y \in \{0,1\}^d$. By a repeated use of the relation $z^2 = z$ for $z \in \{0,1\}$, the function f can be represented as a multilinear polynomial in the $2d$ variables of x and y. By combining terms involving the same monomial in the variables of x, it follows that one can write f as in (1) for an integer R and functions $g_i, h_i : \{0,1\}^d \to \mathbb{F}_p$ such that the g_i's are distinct multilinear monomials of total degree at most $p-1$ in d variables. It thus follows that $R \leq \sum_{i=0}^{p-1}\binom{d}{i}$.

Now, denote by V the vertex set of G and, as before, identify each vertex $X \in V$ with an indicator vector $c_X \in \{0,1\}^d$ in the natural way. We observe that the functions g_i and h_i restricted to V form a functional bi-representation of G over $\mathbb{F}_p$. Indeed, for every two vertices $A, B \in V$ we have $f(c_A,c_B) = 0$ if and only if $|A\cap B| \neq p-1 \pmod p$. If A and B are distinct non-adjacent vertices in G then $|A\cap B| \neq p-1$, since $|A| = |B| = 2p-1$ it follows that $|A\cap B| \neq p-1 \pmod p$ as well, thus $f(c_A,c_B) = 0$. On the other hand, every $A \in V$ satisfies $|A| = 2p-1$, so $|A| = p-1 \pmod p$, and thus $f(c_A,c_A) \neq 0$. By Lemma 12 it follows that $\text{minrk}_p(G) \leq R$, and we are done. ◀

3.2 Orthogonality versus Non-orthogonality

For a prime p and an integer $d \geq 1$, let $G_1(d,p)$ be the graph whose vertex set V consists of the non-self-orthogonal vectors of $\mathbb{F}_p^d$, such that two distinct vertices are adjacent if they are not orthogonal over $\mathbb{F}_p$. The minrank of $G_1(d,p)$ over $\mathbb{F}_p$ is equal to d. For the lower bound, observe that $G_1(d,p)$ contains an independent set of size d. For the upper bound, consider the $|V| \times d$ matrix M over $\mathbb{F}_p$ in which the row indexed by a vertex $v \in V$ is v, and notice that the matrix $M \cdot M^T$ represents $G_1(d,p)$ and that its rank is at most d. Note that variants of the graph $G_1(d,p)$ were found useful in the study of the minrank parameter (see, e.g., [39] and [10, Section 4.1]).

It is natural to consider a variant of $G_1(d,p)$ in which the vertices are replaced by the self-orthogonal vectors of $\mathbb{F}_p^d$ and the edges are defined in the same way. Namely, let $G_2(d,p)$ be the graph whose vertex set consists of the self-orthogonal vectors of $\mathbb{F}_p^d$, such that two distinct vertices are adjacent if they are not orthogonal over $\mathbb{F}_p$. We prove below that in contrast to $G_1(d,p)$ the minrank of $G_2(d,p)$ over $\mathbb{F}_p$, for a fixed p, grows exponentially in d. To this end, we prove the following upper bound on the minrank of its complement. The proof is inspired by an idea used in the context of matching vector codes by Dvir, Gopalan, and Yekhanin [16].

▶ **Proposition 15.** *For every prime p and an integer $d \geq 1$,* $\mathrm{minrk}_p(\overline{G_2(d,p)}) \leq \binom{d+p-2}{p-1} + 1$.

Proof. Let V be the vertex set of $G_2(d,p)$, i.e., the set of self-orthogonal vectors of $\mathbb{F}_p^d$. Let $f : V \times V \to \mathbb{F}_p$ be the function defined by

$$f(x,y) = 1 - \Big(\sum_{i=1}^{d} x_i y_i\Big)^{p-1} \pmod p$$

for every $x, y \in V$. Expanding f as a linear combination of monomials and combining terms involving the same monomial in the variables of x, it follows that f can be written as in (1) for an integer R and functions $g_i, h_i : V \to \mathbb{F}_p$, where $g_1 = 1$ and the g_i's for $i \geq 2$ are distinct monomials of degree exactly $p-1$ in d variables. This yields that $R \leq \binom{d+p-2}{p-1} + 1$.

Now, let us show that the functions g_i and h_i form a functional bi-representation of $\overline{G_2(d,p)}$ over $\mathbb{F}_p$. Indeed, by Fermat's little Theorem, $f(u,v) \neq 0$ if and only if u and v are orthogonal. If u and v are distinct non-adjacent vertices in $\overline{G_2(d,p)}$ then they are not orthogonal, thus $f(u,v) = 0$. On the other hand, by the self-orthogonality of the vectors in V, we have $f(v,v) \neq 0$ for every $v \in V$. By Lemma 12 it follows that $\mathrm{minrk}_p(\overline{G_2(d,p)}) \leq R$, and we are done. ◀

Combining Proposition 15 with Lemma 5 implies the following.

▶ **Corollary 16.** *For every prime p and an integer $d \geq 1$,*

$$\mathrm{minrk}_p(G_2(d,p)) \geq \frac{n}{\binom{d+p-2}{p-1} + 1},$$

where n stands for the number of vertices in $G_2(d,p)$. In particular, using $n \geq p^{d-p+1}$, for a fixed prime p,

$$\mathrm{minrk}_p(G_2(d,p)) \geq p^{(1-o(1))\cdot d}.$$

3.3 A Directed Example

We end this section with a quick application of multivariate polynomials to the minrank of a directed graph. The proof employs an idea of Blokhuis [11] used in the study of the Sperner capacity of the cyclic triangle.

▶ **Proposition 17.** *For an integer $d \geq 1$, let $G = (V,E)$ be the directed graph on $V = \{0,1,2\}^d$ where for every distinct $u, v \in V$, $(u,v) \in E$ if $u - v \in \{0,2\}^d \pmod 3$. Then,* $\mathrm{minrk}_3(G) = 2^d$.

Proof. We start with the upper bound. Let $f : V \times V \to \mathbb{F}_3$ be the function defined by

$$f(x,y) = \prod_{i=1}^{d} (x_i - y_i - 1) \pmod 3$$

for every $x, y \in V$. Expanding f as a linear combination of monomials and combining terms involving the same monomial in the variables of x, it follows that one can write f as in (1) for an integer R and functions $g_i, h_i : V \to \mathbb{F}_p$, where the g_i's are distinct multilinear monomials in d variables. This yields that $R \leq 2^d$.

We now show that the functions g_i and h_i form a functional bi-representation of G over $\mathbb{F}_3$. Indeed, for every distinct vertices $u, v \in V$, if $(u,v) \notin E$ then $u_i - v_i = 1 \pmod 3$ for some i, and thus $f(u,v) = 0$. On the other hand, for every $v \in V$, $f(v,v) = (-1)^d \neq 0$. By Lemma 12 it follows that $\mathrm{minrk}_3(G) \leq R$, as desired.

For the lower bound, observe that the subgraph of G induced by the set of vertices $\{0,1\}^d$ is acyclic. It was shown in [7] that the maximum size of an induced acyclic subgraph forms a lower bound on the minrank parameter, thus the proof is completed. ◀

4 Upper Bounds on Minrank via Inclusion Matrices

In this section we prove upper bounds on the minrank parameter of graphs using the method of higher incidence matrices, more specifically – the class of inclusion matrices. The proofs employ ideas from [21]. We start with a few notations and facts following [6, Chapter 7].

Binomial coefficient polynomials

For an integer $k \geq 0$, the binomial coefficient $\binom{x}{k}$ is a polynomial of degree k over $\mathbb{R}$ defined by

$$\binom{x}{k} = \frac{1}{k!} \cdot x(x-1)\cdots(x-k+1).$$

We say that a polynomial is *integer-valued* if it takes integer values on integers. We need the following fact (see, e.g., [6, Exercise 7.3.3]).

▶ **Fact 18.** *For every $k \geq 0$, the integer-valued polynomials of degree at most k are precisely all the integer linear combinations of the polynomials $\binom{x}{0}, \binom{x}{1}, \ldots, \binom{x}{k}$.*

Inclusion matrices

For integers $d \geq s \geq k \geq 0$, let $N^{(d)}(s,k)$ denote the $\binom{d}{s} \times \binom{d}{k}$ binary matrix, whose rows and columns are indexed by all s-subsets and k-subsets of $[d]$ respectively, defined by

$$(N^{(d)}(s,k))_{A,B} = 1 \text{ if and only if } B \subseteq A$$

for every s-subset A and k-subset B of $[d]$. In addition, let $M^{(d)}(s,k)$ denote the $\binom{d}{s} \times \binom{d}{s}$ integer matrix defined by

$$M^{(d)}(s,k) = N^{(d)}(s,k) \cdot N^{(d)}(s,k)^T. \tag{2}$$

Notice that the entry of $M^{(d)}(s,k)$ indexed by (A,B), where A, B are s-subsets of $[d]$, precisely counts the k-subsets X of $[d]$ that satisfy $X \subseteq A \cap B$. Hence, for every s-subsets A and B of $[d]$,

$$(M^{(d)}(s,k))_{A,B} = \binom{|A \cap B|}{k}. \tag{3}$$

▶ **Lemma 19.** *For every $d \geq s \geq \ell \geq 0$ and $a_0, \ldots, a_\ell \in \mathbb{R}$, the matrix $M = \sum_{k=0}^{\ell} a_k \cdot M^{(d)}(s,k)$ satisfies $\mathrm{rank}_{\mathbb{R}}(M) \leq \binom{d}{\ell}$.*

Proof. We first claim that for every $0 \leq k \leq \ell$, every column of $M^{(d)}(s,k)$ is a linear combination of the columns of $N^{(d)}(s,\ell)$. For $k = \ell$ this follows immediately from (2). To see this for $0 \leq k < \ell$, consider the $\binom{d}{s} \times \binom{d}{k}$ matrix $N^{(d)}(s,\ell) \cdot N^{(d)}(\ell,k)$, and observe that the entry indexed by (A,B) in this matrix, where $A, B \subseteq [d]$, $|A| = s$, $|B| = k$, counts the ℓ-subsets X of $[d]$ that satisfy $B \subseteq X \subseteq A$. If $B \subseteq A$ then the number of these subsets is $\binom{s-k}{\ell-k}$ and otherwise it is 0. It follows that

$$N^{(d)}(s,\ell) \cdot N^{(d)}(\ell,k) = \binom{s-k}{\ell-k} \cdot N^{(d)}(s,k).$$

Hence, every column of $N^{(d)}(s,k)$ is a linear combination of the columns of $N^{(d)}(s,\ell)$, and by (2), the same holds for the columns of $M^{(d)}(s,k)$ where $0 \leq k \leq \ell$. As the matrix M is a linear combination of these matrices, it follows that its columns lie in the space spanned by the columns of $N^{(d)}(s,\ell)$ whose dimension is at most $\binom{d}{\ell}$. This yields the required bound on the rank of M. ◀

4.1 Generalized Kneser Graphs

As our first application of the method of inclusion matrices, we improve the bound given in Proposition 13 for the generalized Kneser graphs $K(d, s, T)$ (recall Definition 8). We note, though, that this improvement is not essential to our application in Section 5.

▶ **Proposition 20.** *For every integers $t \leq s \leq d$, a set $T \subseteq \{0, 1, \ldots, s-1\}$ of size $|T| = t$, and a prime $p > s$,*

$$\mathrm{minrk}_p(K(d, s, T)) \leq \binom{d}{s-t}.$$

Proof. Consider the polynomial $m \in \mathbb{R}[x]$ defined by

$$m(x) = \prod_{j \in \{0,1,\ldots,s-1\} \setminus T} (x - j).$$

Notice that m is an integer-valued polynomial of degree $s - t$. By Fact 18, one can write

$$m(x) = \sum_{k=0}^{s-t} a_k \cdot \binom{x}{k} \tag{4}$$

for integer coefficients $a_0, \ldots, a_{s-t}$. Using the notation in (2), define the $\binom{d}{s} \times \binom{d}{s}$ integer matrix

$$M = \sum_{k=0}^{s-t} a_k \cdot M^{(d)}(s, k),$$

and let $M' = M \pmod p$ for a prime $p > s$. We claim that M' represents the graph $K(d, s, T)$ over $\mathbb{F}_p$. Indeed, using (3) and (4), for every two s-subsets A and B of $[d]$ we have

$$M_{A,B} = \sum_{k=0}^{s-t} a_k \cdot (M^{(d)}(s, k))_{A,B} = \sum_{k=0}^{s-t} a_k \cdot \binom{|A \cap B|}{k} = m(|A \cap B|).$$

Every two distinct vertices A, B non-adjacent in $K(d, s, T)$ satisfy $|A \cap B| \in \{0, 1, \ldots, s-1\} \setminus T$, and thus $M_{A,B} = m(|A \cap B|) = 0$, and in particular $M'_{A,B} = 0$. On the other hand, every vertex A satisfies $|A| = s$, and thus $M_{A,A} = m(s)$, so using the assumption $p > s$ it follows that $M'_{A,A} \neq 0$. Finally, we obtain that

$$\mathrm{minrk}_p(K(d, s, T)) \leq \mathrm{rank}_p(M') \leq \mathrm{rank}_{\mathbb{R}}(M) \leq \binom{d}{s-t},$$

where the second and third inequalities follow from Fact 10 and Lemma 19 respectively, and we are done. ◀

We next consider a modular variant of the generalized Kneser graphs, defined as follows.

▶ **Definition 21.** For integers $t \leq s \leq d$ and q, the graph $K_q(d, s, t)$ is defined as $K(d, s, T)$ where $T = \{i \in \{0, 1, \ldots, s-1\} \mid i = t \pmod q\}$.

The following proposition provides an upper bound on the minrank over $\mathbb{F}_p$ of the graph $K_q(d, s, t)$, where p is a prime and q is a power of p. This bound is crucial for the construction given in Theorem 1, which separates for every *fixed* prime p the ϑ-function of a graph from its minrank over $\mathbb{F}_p$.

▶ **Proposition 22.** *For every prime p, a prime power $q = p^\ell$, and integers $t \le s \le d$ such that $q \le s+1$ and $s = t \pmod q$,*

$$\mathrm{minrk}_p(K_q(d,s,t)) \le \binom{d}{q-1}.$$

Proof. Let p be a prime and let $q = p^\ell$ be a prime power. Consider the polynomial $m \in \mathbb{R}[x]$ defined by $m(x) = \binom{x-t-1}{q-1}$. By Fact 18, m is an integer-valued polynomial of degree $q-1$, which can be written as

$$m(x) = \sum_{k=0}^{q-1} a_k \cdot \binom{x}{k} \tag{5}$$

for integer coefficients $a_0, \ldots, a_{q-1}$. Using the notation in (2), define the $\binom{d}{s} \times \binom{d}{s}$ integer matrix

$$M = \sum_{k=0}^{q-1} a_k \cdot M^{(d)}(s,k),$$

and let $M' = M \pmod p$. We claim that M' represents the graph $K_q(d,s,T)$ over $\mathbb{F}_p$. Indeed, using (3) and (5), for every two s-subsets A and B of $[d]$ we have

$$M_{A,B} = \sum_{k=0}^{q-1} a_k \cdot (M^{(d)}(s,k))_{A,B} = \sum_{k=0}^{q-1} a_k \cdot \binom{|A \cap B|}{k} = m(|A \cap B|).$$

To complete the argument, we need the following fact (see, e.g., [6, Proposition 5.31]).

▶ **Fact 23.** *For every prime p, a prime power $q = p^\ell$ and an integer r, p divides $\binom{r-1}{q-1}$ if and only if q does not divide r.*

By Fact 23 and the definition of m, $M'_{A,B} = 0$ if and only if $|A \cap B| \ne t \pmod q$. Every two distinct vertices A, B non-adjacent in $K_q(d,s,t)$ satisfy $|A \cap B| \ne t \pmod q$, and thus $M'_{A,B} = 0$. On the other hand, every vertex A satisfies $|A| = s$, so using the assumption $s = t \pmod q$, it follows that $M'_{A,A} \ne 0$. Finally, we obtain that

$$\mathrm{minrk}_p(K_q(d,s,t)) \le \mathrm{rank}_p(M') \le \mathrm{rank}_{\mathbb{R}}(M) \le \binom{d}{q-1},$$

where the second inequality follows from Fact 10 and the third follows from Lemma 19 using $q \le s+1$, so we are done. ◀

5 Separations between Minrank and Other Graph Parameters

In this section we prove Theorems 1, 2, and 3. We start with the proof of Theorem 3, which claims the existence of n-vertex graphs whose minrank, over any sufficiently large prime order field, is polynomial in n while their complement is vector 3-colorable. The proof is based on instances of the generalized Kneser graphs, in which pairs of sets are adjacent if their intersection size is small. Such graphs were used in [31] to provide a similar separation between the vector chromatic number and the chromatic number.

Proof of Theorem 3. For a sufficiently large integer t define $d = 8t$, $s = 4t$, and $T = \{0, 1, \ldots, t\}$. Let G be the complement of the graph $K(d,s,T)$ given in Definition 8, and note that the number n of its vertices satisfies $n = \binom{8t}{4t} = 2^{(1-o(1))d}$. By Item 1 of Lemma 9,

$\chi_v(\overline{G}) \leq \frac{d(s-t)}{s^2-dt} = 3$. Apply Proposition 20 to get that for any prime $p > s = \Theta(\log n)$, we have

$$\text{minrk}_p(\overline{G}) = \text{minrk}_p(K(d,s,T)) \leq \binom{d}{s-|T|} = \binom{8t}{3t-1}.$$

By Lemma 5, this implies that

$$\text{minrk}_p(G) \geq \frac{n}{\binom{8t}{3t-1}} \geq n^{1-H(3/8)-o(1)} \geq n^{0.0455},$$

where H stands for the binary entropy function. This completes the proof. ◀

We turn to prove Theorem 1, which claims that for every fixed prime p there exist n-vertex graphs with constant ϑ-function and minrank over $\mathbb{F}_p$ polynomial in n. Here we use the modular variant of the generalized Kneser graphs considered in Proposition 22 (recall Definition 21). For $p = 2$, our graphs are related to a construction of [5].

Proof of Theorem 1. We first prove the theorem for $p = 2$. For a sufficiently large integer ℓ, let $d = 2^\ell$, $q = \frac{d}{4}$, $t = \frac{d}{8}$, and $s = t + q = \frac{3}{8} \cdot d$. Let G be the complement of the graph $K_q(d,s,t)$ given in Definition 21, and let $n = \binom{d}{s}$ denote the number of its vertices. Recall that two distinct vertices A and B, representing s-subsets of $[d]$, are adjacent in $K_q(d,s,t)$ if and only if $|A \cap B| = t \pmod q$. By $0 \leq t < q$ and $s = t + q$, this condition is equivalent for distinct A and B to $|A \cap B| = t$, so in our setting $K_q(d,s,t) = K(d,s,\{t\})$. Recalling that $\vartheta(G)$ is equal to the strict vector chromatic number of $\overline{G}$, by Item 2 of Lemma 9 we obtain that

$$\vartheta(G) = \chi_v^{(s)}(K(d,s,\{t\})) \leq \frac{d(s-t)}{s^2-dt} = \frac{1/4}{(3/8)^2 - 1/8} = 16.$$

Now, as q is a power of 2 and $s = t \pmod q$, we can apply Proposition 22 to obtain that

$$\text{minrk}_2(\overline{G}) = \text{minrk}_2(K_q(d,s,t)) \leq \binom{d}{q-1} \leq 2^{H(1/4)\cdot d},$$

where H stands for the binary entropy function. By Lemma 5, it follows that

$$\text{minrk}_2(G) \geq \frac{n}{2^{H(1/4)\cdot d}} \geq n^{1-\frac{H(1/4)}{H(3/8)}-o(1)} \geq n^{0.1499},$$

and we are done.

The proof for a general prime $p \geq 3$ is similar. Details follow. For a sufficiently large integer ℓ, let $d = p^\ell$, $q = \frac{d}{p}$, $t = \frac{d}{p^2}$, and $s = t + q = \frac{p+1}{p^2} \cdot d$. As in the case of $p = 2$, let G be the complement of the graph $K_q(d,s,t) = K(d,s,\{t\})$, and let $n = \binom{d}{s}$ denote the number of its vertices. By Item 2 of Lemma 9 we obtain that

$$\vartheta(G) = \chi_v^{(s)}(K(d,s,\{t\})) \leq \frac{d(s-t)}{s^2-dt} = \frac{\frac{1}{p}}{\frac{(p+1)^2}{p^4} - \frac{1}{p^2}} = \frac{p^3}{2p+1}.$$

In particular, $\vartheta(G) \leq c$ for some $c = c(p)$. As q is a power of the prime p and $s = t \pmod q$, we can apply Proposition 22 to obtain that

$$\text{minrk}_p(\overline{G}) = \text{minrk}_p(K_q(d,s,t)) \leq \binom{d}{q-1} \leq 2^{H(1/p)\cdot d}.$$

By Lemma 5, using $p \geq 3$ and the monotonicity of H in $[0, 0.5]$, it follows that

$$\text{minrk}_p(G) \geq \frac{n}{2^{H(1/p)\cdot d}} \geq n^{1-\frac{H(1/p)}{H((p+1)/p^2)}-o(1)} \geq n^\delta,$$

for some $\delta = \delta(p) > 0$, completing the proof. ◀

Finally, we prove the following theorem that confirms Theorem 2. Here we use the generalized Kneser graphs considered in Proposition 14.

▶ **Theorem 24.** *For any $\delta < 0.1887$ there exists $c = c(\delta)$ such that for infinitely many integers n there exists an n-vertex graph G such that $\vartheta(G) \leq c$ and $\mathrm{minrk}_p(G) \geq n^\delta$ for some $p = \Theta(\log n)$.*

Proof. For a sufficiently large prime p, let $\varepsilon \in (0,2)$ be a real number such that $d = (4-\varepsilon)\cdot p$ is an integer, and let $s = 2p-1$ and $t = p-1$. Let G be the complement of the graph $K(d,s,\{t\})$. Since $s^2 > dt$ we can apply Item 2 of Lemma 9 to obtain that

$$\begin{aligned}\vartheta(G) &= \chi_v^{(s)}(\overline{G}) = \chi_v^{(s)}(K(d,s,\{t\})) \leq \frac{d(s-t)}{s^2-dt} \\ &= \frac{(4-\varepsilon)p^2}{(2p-1)^2-(4-\varepsilon)p(p-1)} = \frac{(4-\varepsilon)p^2}{\varepsilon p^2-\varepsilon p+1} \leq \frac{(4-\varepsilon)p^2}{\varepsilon p^2/2} \leq \frac{2(4-\varepsilon)}{\varepsilon},\end{aligned}$$

where in the second inequality we have used the assumption that p is sufficiently large. Now, by Proposition 14 it follows that

$$\mathrm{minrk}_p(\overline{G}) = \mathrm{minrk}_p(K(d,s,\{t\})) \leq \sum_{i=0}^{p-1}\binom{d}{i} \leq 2^{H(1/(4-\varepsilon))\cdot d}.$$

Let n denote the number of vertices in G, and notice that $n = \binom{d}{2p-1} = \binom{d}{d-2p+1}$. Applying Lemma 5, we get that

$$\mathrm{minrk}_p(G) \geq \frac{n}{2^{H(1/(4-\varepsilon))\cdot d}} \geq n^{1-\frac{H(1/(4-\varepsilon))}{H((2-\varepsilon)/(4-\varepsilon))}-o(1)},$$

where $p = \Theta(d) = \Theta(\log n)$.

Finally, notice that for every $\delta < 1 - H(1/4) \approx 0.1887$ one can choose a sufficiently small $\varepsilon > 0$ for which the above construction gives an n-vertex graph G with $\vartheta(G) \leq c$ and $\mathrm{minrk}_p(G) \geq n^\delta$, where c depends only on δ and $p = \Theta(\log n)$. ◀

References

1. Rudolf Ahlswede, Ning Cai, Shuo-Yen Robert Li, and Raymond W. Yeung. Network information flow. *IEEE Trans. Inform. Theory*, 46(4):1204–1216, 2000.
2. Noga Alon. The Shannon capacity of a union. *Combinatorica*, 18(3):301–310, 1998.
3. Noga Alon, László Babai, and H. Suzuki. Multilinear polynomials and Frankl–Ray-Chaudhuri–Wilson type intersection theorems. *J. Comb. Theory, Ser. A*, 58(2):165–180, 1991.
4. Noga Alon and Nabil Kahale. Approximating the independence number via the ϑ-function. *Math. Program.*, 80:253–264, 1998.
5. Noga Alon and Yuval Peres. A note on Euclidean Ramsey theory and a construction of Bourgain. *Acta Math. Hungar.*, 57(1-2):61–64, 1991.
6. László Babai and Peter Frankl. *Linear Algebra Methods in Combinatorics With Applications to Geometry and Computer Science*. Wiley-Interscience Series in Discrete Mathematics and Optimization. The University of Chicago, second edition, 1992.
7. Ziv Bar-Yossef, Yitzhak Birk, T. S. Jayram, and Tomer Kol. Index coding with side information. In *FOCS*, pages 197–206, 2006.
8. Piotr Berman and Georg Schnitger. On the complexity of approximating the independent set problem. *Inf. Comput.*, 96(1):77–94, 1992. Preliminary version in STACS'89.

9 Yitzhak Birk and Tomer Kol. Coding on demand by an informed source (ISCOD) for efficient broadcast of different supplemental data to caching clients. *IEEE Trans. Inform. Theory*, 52(6):2825–2830, 2006. Preliminary version in INFOCOM'98.

10 Anna Blasiak, Robert Kleinberg, and Eyal Lubetzky. Broadcasting with side information: Bounding and approximating the broadcast rate. *IEEE Trans. Information Theory*, 59(9):5811–5823, 2013.

11 Aart Blokhuis. On the Sperner capacity of the cyclic triangle. *Journal of Algebraic Combinatorics*, 2(2):123–124, 1993.

12 Moses Charikar. On semidefinite programming relaxations for graph coloring and vertex cover. In *Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 616–620, 2002.

13 Eden Chlamtáč and Ishay Haviv. Linear index coding via semidefinite programming. *Combinatorics, Probability & Computing*, 23(2):223–247, 2014. Preliminary version in SODA'12.

14 Amin Coja-Oghlan. The Lovász number of random graphs. *Combinatorics, Probability & Computing*, 14(4):439–465, 2005. Preliminary version in RANDOM'03.

15 Irit Dinur and Igor Shinkar. On the conditional hardness of coloring a 4-colorable graph with super-constant number of colors. In *APPROX-RANDOM*, pages 138–151, 2010.

16 Zeev Dvir, Parikshit Gopalan, and Sergey Yekhanin. Matching vector codes. *SIAM J. Comput.*, 40(4):1154–1178, 2011. Preliminary version in FOCS'10.

17 Salim El Rouayheb, Alex Sprintson, and Costas Georghiades. On the relation between the index coding and the network coding problems. In *proceedings of IEEE International Symposium on Information Theory*, pages 1823–1827. IEEE Press, 2008.

18 Uriel Feige. Randomized graph products, chromatic numbers, and the Lovász ϑ-function. *Combinatorica*, 17(1):79–90, 1997. Preliminary version in STOC'95.

19 Uriel Feige, Michael Langberg, and Gideon Schechtman. Graphs with tiny vector chromatic numbers and huge chromatic numbers. *SIAM J. Comput.*, 33(6):1338–1368, 2004. Preliminary version in FOCS'02.

20 Peter Frankl and Vojtěch Rödl. Forbidden intersections. *Trans. Amer. Math. Soc.*, 300(1):259–286, 1987.

21 Peter Frankl and Richard M. Wilson. Intersection theorems with geometric consequences. *Combinatorica*, 1(4):357–368, 1981.

22 Alexander Golovnev, Oded Regev, and Omri Weinstein. The minrank of random graphs. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2017*, pages 46:1–46:13, 2017.

23 Martin Grötschel, László Lovász, and Alexander Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981.

24 Willem Haemers. On some problems of Lovász concerning the Shannon capacity of a graph. *IEEE Trans. Inform. Theory*, 25(2):231–232, 1979.

25 Willem Haemers. An upper bound for the Shannon capacity of a graph. In *Algebraic methods in graph theory, Vol. I, II (Szeged, 1978)*, volume 25 of *Colloq. Math. Soc. János Bolyai*, pages 267–272. North-Holland, Amsterdam, 1981.

26 Ishay Haviv and Michael Langberg. On linear index coding for random graphs. In *IEEE International Symposium on Information Theory*, pages 2231–2235, 2012.

27 Ishay Haviv and Michael Langberg. H-wise independence. In *Innovations in Theoretical Computer Science (ITCS'13)*, pages 541–552, 2013.

28 Syed A. Jafar. Topological interference management through index coding. *IEEE Transactions on Information Theory*, 60(1):529–568, 2014.

29 Mingyue Ji, Antonia M. Tulino, Jaime Llorca, and Giuseppe Caire. Caching and coded multicasting: Multiple groupcast index coding. In *IEEE Global Conference on Signal and Information Processing*, pages 881–885, 2014.

30 Jeff Kahn and Gil Kalai. A counterexample to Borsuk's conjecture. *Bull. Amer. Math. Soc.*, 29(1):60–62, 1993.

31 David R. Karger, Rajeev Motwani, and Madhu Sudan. Approximate graph coloring by semidefinite programming. *J. ACM*, 45(2):246–265, 1998. Preliminary version in FOCS'94.

32 Donald E. Knuth. The sandwich theorem. *Electronic J. Combinatorics*, 1:1, 1994.

33 Michael Langberg and Alexander Sprintson. On the hardness of approximating the network coding capacity. *IEEE Transactions on Information Theory*, 57(2):1008–1014, 2011. Preliminary version in ISIT'08.

34 László Lovász. On the Shannon capacity of a graph. *IEEE Trans. Inform. Theory*, 25(1):1–7, 1979.

35 Eyal Lubetzky and Uri Stav. Nonlinear index coding outperforming the linear optimum. *IEEE Trans. Inform. Theory*, 55(8):3544–3551, 2009. Preliminary version in FOCS'07.

36 Hamed Maleki, Viveck R Cadambe, and Syed A Jafar. Index coding - An interference alignment perspective. *IEEE Transactions on Information Theory*, 60(9):5402–5432, 2014.

37 Arya Mazumdar. On a duality between recoverable distributed storage and index coding. In *IEEE International Symposium on Information Theory*, pages 1977–1981, 2014.

38 Eran Ofek and Uriel Feige. Random 3CNF formulas elude the Lovász theta function. *Electronic Colloquium on Computational Complexity (ECCC)*, 13(043), 2006.

39 René Peeters. Orthogonal representations over finite fields and the chromatic number of graphs. *Combinatorica*, 16(3):417–431, 1996.

40 Pavel Pudlák, Vojtech Rödl, and Jirí Sgall. Boolean circuits, tensor ranks, and communication complexity. *SIAM J. Comput.*, 26(3):605–633, 1997.

41 Dijen K. Ray-Chaudhuri and Richard M. Wilson. On t-designs. *Osaka J. Math.*, 12(3):737–744, 1975.

42 Søren Riis. Information flows, graphs and their guessing numbers. *Electr. J. Comb.*, 14(1), 2007.

43 Claude E. Shannon. The zero error capacity of a noisy channel. *Institute of Radio Engineers, Transactions on Information Theory*, IT-2:8–19, 1956.

44 Leslie G. Valiant. Why is Boolean complexity theory difficult? In *Poceedings of the London Mathematical Society symposium on Boolean function complexity*, volume 169, pages 84–94, 1992.

Online Makespan Minimization: The Power of Restart

Zhiyi Huang[1]
Department of Computer Sicence, The University of Hong Kong, Hong Kong
zhiyi@cs.hku.hk

Ning Kang
Department of Computer Sicence, The University of Hong Kong, Hong Kong
nkang@cs.hku.hk

Zhihao Gavin Tang
Department of Computer Sicence, The University of Hong Kong, Hong Kong
zhtang@cs.hku.hk

Xiaowei Wu[2]
Department of Computing, The Hong Kong Polytechnic University, Hong Kong
wxw0711@gmail.com

Yuhao Zhang
Department of Computer Sicence, The University of Hong Kong, Hong Kong
yhzhang2@cs.hku.hk

Abstract

We consider the online makespan minimization problem on identical machines. Chen and Vestjens (ORL 1997) show that the largest processing time first (LPT) algorithm is 1.5-competitive. For the special case of two machines, Noga and Seiden (TCS 2001) introduce the SLEEPY algorithm that achieves a competitive ratio of $(5-\sqrt{5})/2 \approx 1.382$, matching the lower bound by Chen and Vestjens (ORL 1997). Furthermore, Noga and Seiden note that in many applications one can kill a job and restart it later, and they leave an open problem whether algorithms with restart can obtain better competitive ratios.

We resolve this long-standing open problem on the positive end. Our algorithm has a natural rule for killing a processing job: a newly-arrived job replaces the smallest processing job if 1) the new job is larger than other pending jobs, 2) the new job is much larger than the processing one, and 3) the processed portion is small relative to the size of the new job. With appropriate choice of parameters, we show that our algorithm improves the 1.5 competitive ratio for the general case, and the 1.382 competitive ratio for the two-machine case.

2012 ACM Subject Classification Theory of computation → Scheduling algorithms, Theory of computation → Approximation algorithms analysis, Theory of computation → Online algorithms

Keywords and phrases Online Scheduling, Makespan Minimization, Identical Machines

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.14

Related Version A full version of the paper can be found at [19], `https://arxiv.org/abs/1806.02207`.

[1] Partially supported by the Hong Kong RGC under the grant HKU17202115E.
[2] Part of the work was done when the author was a postdoc at the University of Hong Kong.

1 Introduction

We study in this paper the classic online scheduling problem on identical machines. Let there be m identical machines, and a set of jobs that arrive over time. For each job j, let r_j denote its release time (arrival time), and p_j denote its processing time (size). We assume without loss of generality that all r_j's and p_j's are distinct. We seek to schedule each job on one of the m machines such that the makespan (the completion time of the job that completes last) is minimized.

We adopt the standard assumption that there is a pending pool such that jobs released but not scheduled are in the pending pool. That is, the algorithm does not need to assign a job to one of the machines at its arrival; it can decide later when a machine becomes idle. Alternatively, the *immediate-dispatching* model has also been considered in some papers (e.g., [2]). We consider the standard competitive analysis of online algorithms. An algorithm is $(1+\gamma)$-competitive if for any online sequence of jobs, the makespan of the schedule made by the algorithm is at most $(1+\gamma)$ times the minimum makespan in hindsight. Without loss of generality, (by scaling the job sizes) we assume the minimum makespan $\mathsf{OPT} = 1$ (for analysis purpose only).

Chen and Vestjens [8] consider a greedy algorithm called *largest processing time first* (LPT): whenever there is an idle machine, schedule the largest job in the pending pool. They prove that the LPT algorithm is 1.5-competitive and provide a matching lower bound (consider m jobs of size 0.5 followed by a job of size 1). They also show that no online algorithm can achieve a competitive ratio better than 1.3473. For the special case when there are only two machines, Noga and Seiden [25] introduce the SLEEPY algorithm that achieves a tight $(5-\sqrt{5})/2 \approx 1.382$ competitive ratio, due to a previous lower bound given by Chen and Vestjens [8].

The 1.382 lower bound (for two machines) holds under the assumption that whenever a job is scheduled, it must be processed all the way until its completion. However, as noted in [25], many applications allow **restart**: a job being processed can be killed (put into pending) and restarted later to make place for a newly-arrived job; a job is considered completed only if it has been continuously processed on some machine for a period of time that equals to its size. In other words, whenever a job gets killed, all previous processing of this job is wasted.

Note that the restart setting is different from the *preemptive* setting, in which the processed portion is not wasted. Noga and Seiden [25] leave the following open problem: *Is it possible to beat the* 1.382 *barrier with restart?*

In this paper, we bring an affirmative answer to this long-standing open problem.

We propose a variant of the LPT algorithm (with restart) that improves the 1.5 competitive ratio for the general case, and the 1.382 competitive ratio for the two-machine case.

Our Replacement Rule. A naïve attempt for the replacement rule would be to replace a job whenever the newly-arrived job has a larger size. However, it is easy to observe that the naïve attempt fails even on one machine: the worst case competitive ratio is 2 if we keep replacing jobs that are almost completed (with jobs of slightly larger size). Hence we should prevent a job from being replaced if a large portion has been processed. Moreover, we allow a newly-arrived job to replace a processing job only if it has a much larger size, in order to avoid a long chain of replacements. As we will show by an example in Section C, the worst case competitive ratio is 1.5 if a job of size 1 is replaced by a job of size $1+\epsilon$, which is in turn replaced by a job of size $1+2\epsilon$, etc.

We hence propose the following algorithm that applies the above rules.

LPT with Restart. As in the LPT algorithm, our algorithm schedules the largest pending job whenever there is an idle machine. The main difference is that our algorithm may kill a processing job to make place for a newly-arrived job according to the following rule. Upon the arrival of a job j, we kill a processing job k (i.e., put k into pending) and schedule j if:
1. j is the largest pending job and k is the smallest among the m processing jobs;
2. the processed portion of k is less than αp_j;
3. the size of j is more than $1+\beta$ times larger than k, i.e., $p_j > (1+\beta)p_k$,

where $0 < \alpha, \beta < \frac{1}{2}$ are parameters of the algorithm. We call such an operation a *replacement* (i.e. j replaces k).

Intuitively, the parameter α provides a bound on the total amount of wasted processing (in terms of the total processing time); while the parameter β guarantees an exponential growth in the processing time of jobs if there is a chain of replacements. With appropriate choice of parameters, we show the following results.

▶ **Theorem 1.** *LPT with Restart, with parameters $\alpha = \frac{1}{200}$ and $\beta = \sqrt{2}-1$, is $(1.5 - \frac{1}{20000})$-competitive for the Online Makespan Minimization problem with restart.*

▶ **Theorem 2.** *LPT with Restart, with parameters $\alpha = \beta = 0.2$, is 1.38-competitive for the Online Makespan Minimization problem with restart on two machines.*

There are many other natural candidate replacement rules. We list some candidate algorithms that we have considered and their counter examples in Section C.

Our Techniques. The main focus of our paper is the general case, i.e., on m machines. The analysis for the two-machine case is built on the general case by refining some of the arguments. We adopt an idea from Noga and Seiden [8] to look at the last completed job in our schedule. Intuitively, only jobs with size comparable to that of the last job matter. We develop two kinds of arguments, namely the *bin-packing argument* and the *efficiency argument*.

Assume for contrary that the algorithm has a makespan strictly larger than $1+\gamma$, where $\gamma := \frac{1}{2} - \frac{1}{20000}$, we use the bin-packing argument to give an upper bound on the size of the last completed job. Assume that the last job is large, we will find a number of large jobs that cannot be packed into m bins of size 1 (recall that we assume $\mathsf{OPT} = 1$). In other words, to schedule this set of jobs, one of the m machines must get a total workload strictly greater than 1. For example, finding $2m+1$ jobs of size strictly greater than $\frac{1}{3}$ would suffice. We refer to such a set of large jobs as an *infeasible* set of jobs.

We then develop an efficiency argument to handle the case when the last job is of small size. The central of the argument is a Leftover Lemma that upper bounds the difference of total processing done by the algorithm and by OPT. As our main technical contribution, the lemma is general enough to be applied to all schedules.

Fix any schedule (produced by some algorithm ALG) and a time t. Let $\mathcal{M}$ denote the set of machines. For each machine $M \in \mathcal{M}$, let $\mathsf{W}(M,x) \in \{0,1\}$ be the indicator function of the event that "at time x, machine M is not processing while there are pending jobs". Define $\mathsf{W}_t = \sum_{M\in\mathcal{M}} \int_0^t \mathsf{W}(M,x)dx$ to be the total waste (of processing power) before time t. We show (in Section 3) the following lemma that upper bounds the leftover workload.

▶ **Lemma 3** (Leftover Lemma). *For all time t, let Δ_t be the difference in total processing time before time t between ALG and OPT. We have $\Delta_t \le \frac{1}{4}tm + \mathsf{W}_t$.*

Observe that the total processing power (of m machines) before time t is tm. The Leftover Lemma says that compared to any schedule (produced by algorithm ALG), the *extra* processing the optimal schedule can finish before time t, is upper bounded by the

processing power wasted by the schedule (e.g., due to replacements), plus a quarter of the total processing power, which comes from the sub-optimal schedule of jobs.

Consider applying the lemma to the final schedule[3] produced by our algorithm. Since our algorithm schedules a job whenever a machine becomes idle, the waste W_t comes only from the processing (before time t) of jobs that are replaced. Thus (by our replacement) we can upper bound W_t by α fraction of the total size of jobs that replace other jobs.

We remark that the above bound on the leftover workload is tight for LPT (for which $\mathsf{W}_t = 0$). Consider m jobs of size 0.5 arriving at time 0, followed by $m/2$ jobs of size $1-\epsilon$ arriving at time ϵ. The optimal schedule uses $\frac{m}{2}$ machines to process the size $(1-\epsilon)$ jobs and $\frac{m}{2}$ machines to process the size 0.5 jobs (two per machine), finishing all jobs at time 1. LPT would schedule all the size 0.5 jobs first; all of the $\frac{m}{2}$ size $(1-\epsilon)$ jobs have half of their workload unprocessed at time 1. Therefore, the amount of leftover workload at time 1 is $\frac{m}{4}$.

Other Work. The online scheduling model with restart has been investigated in the problem of scheduling jobs on a single machine to maximize the number of jobs completed before their deadlines. Hoogeveen et al. [18] study the general case and propose a 2-competitive algorithm with restart. Subsequently, Chrobak et al. [9] consider the special case when jobs have equal lengths. They propose an improved $\frac{3}{2}$-competitive algorithm with restart for this special case, and prove that this is optimal for deterministic algorithms. However, the restart rule and its analysis in our paper do not bear any obvious connections to those in [18] and [9] due to the different objectives.

Other settings of the online makespan minimization problem have been studied in the literature. A classic setting is when all machines are identical and all jobs have release time 0, but the algorithm must immediately assign each job to one of the machines at its arrival (immediate dispatching). This is the same as online load balancing problem. Graham [17] proves that the natural greedy algorithm that assigns jobs to the machine with the smallest workload is $(2-\frac{1}{m})$-competitive in this setting, which is optimal for $m \leq 3$ (due to folklore examples). A series of research efforts have then been devoted to improving the competitive ratio when m is large (e.g., [1, 3, 22]). For $m = 4$, the best upper bound is 1.7333 [7], while the best lower bound stands at 1.7321 [21]. For m that tends to infinity, the best upper bound is 1.9201 [16], while the best lower bound is 1.880 [20]. A variant of the above setting is that a buffer is provided for temporarily storing a number of jobs; when the buffer is full, one of the jobs must be removed from the buffer and allocated to a machine (e.g., [24, 10]). Kellerer et al. [23] and Zhang [27] use algorithms with a buffer of size one to achieve an improved 4/3 competitive ratio for two machines. Englert et al. [13] characterize the best ratio achievable with a buffer of size $\Theta(m)$, where the ratio is between 4/3 and 1.4659 depending on the number of machines m. When both preemption and migration are allowed, Chen et al. [6] give a 1.58-competitive algorithm without buffer, matching the previous lower bound by Chen et al. [5]. Dósa and Epstein [11] achieve a ratio of 4/3 with a buffer of size $\Theta(m)$.

Finally, if the machines are related instead of identical, the best known algorithm is 4.311-competitive by Berman et al. [4], while the best lower bound is 2 by Epstein and Sgall [15]. When preemption is allowed, Ebenlendr et al. [12] show that the upper bound can be improved to e. For the special case of two related machines, the current best competitive ratio is 1.53 by Epstein et al. [14] without preemption, and 4/3 with preemption by Ebenlendr et al. [12] and Wen and Du [26].

[3] Since a job can be scheduled and replaced multiple times, its start time is finalized only when it is completed.

Organization. We first provide some necessary definitions in Section 2. Then we prove the most crucial structural property (Lemma 3, the Leftover Lemma) in Section 3, which essentially gives a lower bound on the efficiency of all schedules. We present the details of the bin-packing argument and efficiency argument in Section 4, where our main result Theorem 1 is proved. For space reasons, the analysis for the special case of two machines is omitted. Interested readers can refer to the full version of our paper [19]. Finally, we prove in Appendix D that no deterministic algorithm, even with restart, can get a competitive ratio better than $\sqrt{1.5} \approx 1.225$.

2 Preliminaries

Consider the online makespan minimization with m identical machines and jobs arriving over time. Recall that for each job j, r_j denotes its release time and p_j denotes its size. Let OPT and ALG be the makespan of the optimal schedule and our schedule, respectively. Recall that we assume without loss of generality that $\mathsf{OPT} = 1$ (for analysis purpose only). Hence we have $r_j + p_j \leq 1$ for all jobs j. Further, let s_j and $c_j := s_j + p_j$ denote the start and completion time of job j, respectively, in the **final** schedule produced by our online algorithm. Note that a job can be scheduled and replaced multiple times. We use $s_j(t)$ to denote the last start time of j before time t.

We use n to denote the job that completes last, i.e., we have $\mathsf{ALG} = c_n = s_n + p_n$.

We consider the time horizon as continuous, and starts from $t = 0$. W.l.o.g. (by perturbing the variables slightly), we assume that all r_i's, p_i's and s_i's are different.

▶ **Definition 4** (Processing Jobs)**.** For any $t \leq \mathsf{ALG}$, we denote by $J(t)$ the set of jobs that are being processed at time t, including the jobs that are completed or replaced at t but excluding the jobs that start at t.

Note that $J(t)$ is defined based on the schedule produced by the algorithm at time t. It is possible that jobs in $J(t)$ are replaced at or after time t.

Idle and Waste. We say that a machine is *idle* in time period (a, b), if for all $t \in (a, b)$, the machine is not processing any job according to our algorithm, and there is **no** pending job. We call time t idle if there exists at least one idle machine at time t. Whenever a job k is replaced by a job j (at r_j), we say that a *waste* is created at time r_j. The size of the waste is the portion of k that is (partially) processed before it is replaced. We can also interpret the waste as a time period on the machine. We say that the waste *comes from* k, and call j the *replacer*.

▶ **Definition 5** (Total Idle and Total Waste)**.** For any $t \in [0, 1]$, define I_t as the *total idle time* before time t, i.e., the summation of total idle time before time t on each machine. Similarly, define W_t as the *total waste* before time t in the final schedule, i.e., the total size of wastes located before time t, where if a waste crosses t, then we only count its fractional size in $[0, t]$.

3 Bounding Leftover: Idle and Waste

In this section, we prove Lemma 3, the most crucial structural property. Recall that we define W_t as the total waste located before time t. For applying the lemma to general scheduling algorithms, (recall from Section 1) W_t is defined as $\sum_{M \in \mathcal{M}} \int_0^t \mathsf{W}(M, x)dx$, the total time during which machines are not processing while there are pending jobs. It is easy to check that the proofs hold under both definitions. We first give a formal definition of the *leftover* Δ_t at time t.

▶ **Definition 6** (Leftover)**.** Consider the final schedule and a fixed optimal schedule OPT. For any $t \in [0,1]$, let Δ_t be the total processing OPT does before time t, minus the total processing our algorithm does before time t.

Since the optimal schedule can process a total processing at most $m(1-t)$ after time t, we have the following useful observation.

▶ **Observation 3.1.** *The total processing our algorithm does after time t is at most $m(1-t)+\Delta_t$.*

We call time t a *marginal idle time* if t is idle and the time immediately after t is not. We first define A_t, which is designated to be an upper bound on the total processing that could have been done before time t, i.e., the leftover workload due to sub-optimal schedule.

▶ **Definition 7** (A_t)**.** For all $t \in [0,1]$, if there is no idle time before t, then define $\mathsf{A}_t = 0$, otherwise let $t' \le t$ be the last idle time before t. Define $\mathsf{A}_t = \sum_{j \in J(t')} \min\{\delta_j, p_j\}$, where $\delta_j := |T_j| = |\{\theta \in [r_j, t'] : \text{job } j \text{ is pending at time } \theta\}|$ is the total pending time of job $j \in J(t')$ before time t'.

We show the following claim, which (roughly) says that the extra processing OPT does (compared to ALG) before time t, is not only upper bounded by total idle and waste ($\mathsf{I}_t + \mathsf{W}_t$), but also by the total size or pending time of jobs currently being processed ($\mathsf{A}_t + \mathsf{W}_t$).

▶ **Claim 3.1.** *We have $\Delta_t \le \min\{\mathsf{A}_t, \mathsf{I}_t\} + \mathsf{W}_t$ for all $t \in [0,1]$.*

Proof. First observe that we only need to prove the claim for marginal idle times, as we have $\frac{d\Delta_t}{dt} \le \frac{d\mathsf{W}_t}{dt}$ (while $\frac{d\mathsf{A}_t}{dt} = \frac{d\mathsf{I}_t}{dt} = 0$) for non-idle time t. Now suppose t is a marginal idle time.

It is easy to see that Δ_t is at most $\mathsf{I}_t + \mathsf{W}_t$, the total length of time periods before t during which the algorithm is not processing (in the final schedule). Next we show that $\Delta_t \le \mathsf{A}_t + \mathsf{W}_t$.

Let $\Delta_t(t)$, $\mathsf{A}_t(t)$ and $\mathsf{W}_t(t)$ be the corresponding variables when the algorithm is run until time t. Observe that for a job $j \in J(t)$, if it is replaced after time t, then it contributes a waste to W_t but not to $\mathsf{W}_t(t)$. Moreover, it has the same contribution to $\Delta_t - \Delta_t(t)$ and to W_t. Thus we have $\mathsf{W}_t - \mathsf{W}_t(t) = \Delta_t - \Delta_t(t)$. By definition we have $\mathsf{A}_t = \mathsf{A}_t(t)$.

Hence it suffices to show that $\Delta_t(t) \le \mathsf{A}_t + \mathsf{W}_t(t)$.

Since t is idle, there is no pending job at time t. Thus the difference in total processing at time t, i.e., Δ_t, must come from the difference (between ALG and OPT) in processing of jobs in $J(t)$ that has been completed. For each $j \in J(t)$, the extra processing OPT can possibly do on j (compared to ALG) is at most $\min\{s_j(t) - r_j, p_j\}$. Hence we have $\Delta_t(t) \le \sum_{j \in J(t)} \min\{s_j(t) - r_j, p_j\}$.

Recall by Definition 7, we have $T_j \subset [r_j, s_j(t))$ is the periods during which j is pending.

Thus at every time $t \in [r_j, s_j(t)) \setminus T_j$, j is being processed (and replaced later). Hence $\big|[r_j, s_j(t)) \setminus T_j\big|$ is at most the total wastes from j that are created before $s_j(t) < t$, which implies

$$\Delta_t(t) \le \sum_{j \in J(t)} \min\{s_j(t) - r_j, p_j\} \le \sum_{j \in J(t)} \min\{\delta_j, p_j\} + \mathsf{W}_t(t) = \mathsf{A}_t + \mathsf{W}_t(t),$$

as desired. ◀

We prove the following technical claim.

▶ **Claim 3.2.** *For any integer $k \ge 1$, given any three sequences of positive reals $\{a_i\}_{i \in [k]}$, $\{b_i\}_{i \in [k]}$ and $\{h_i\}_{i \in [k]}$ satisfying conditions*

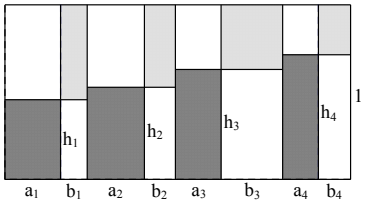


Figure 1 graph representation of Claim 3.2 for $k = 4$.

(1) $0 \le h_1 \le h_2 \le \ldots \le h_k \le 1$;
(2) *for all* $j \in [k]$, *we have* $\sum_{i\in[j]} a_i h_i \ge \frac{1}{4}\sum_{i\in[j]}(a_i + b_i)$, *we have* $\sum_{i\in[k]} b_i(1 - h_i) \le \frac{1}{4}\sum_{i\in[k]}(a_i + b_i)$.

Proof. We prove the claim by induction on k. We first show that the claim holds true when $k = 1$. Note that we have $a_1h_1 \cdot b_1(1 - h_1) \le (\frac{a_1+b_1}{2})^2 \cdot (\frac{h_1+(1-h_1)}{2})^2 = \frac{(a_1+b_1)^2}{16}$. Combine with property (2) we know that $b_1(1 - h_1) \le \frac{1}{4}(a_1 + b_1)$.

Now suppose the claim is true for all values smaller than k. Using induction hypothesis on $\{a_i\}_{i\in[k-1]}, \{b_i\}_{i\in[k-1]}$ and $\{h_i\}_{i\in[k-1]}$, we have

$$\sum_{i\in[k-1]} b_i(1 - h_i) \le \frac{1}{4}\sum_{i\in[k-1]}(a_i + b_i) \le \sum_{i\in[k-1]} a_ih_i.$$

Define $\phi = \min\{b_k, \sum_{i\in[k-1]}(4a_i \cdot h_i - a_i - b_i)\}$. Let $b'_k = b_k - \phi$ and $b'_{k-1} = b_{k-1} + \phi$.

Note that $\{a_i\}_{i\in[k]}, \{b_i\}_{i\in[k-2]} \cup \{b'_{k-1}, b'_k\}$ and $\{h_i\}_{i\in[k]}$ (and their prefixes) satisfy the conditions of the claim: first, by definition we have $b'_k > 0$ and $b'_{k-1} \ge b_{k-1} > 0$; second, since $\{a_i\}_{i\in[k]}$ and $\{h_i\}_{i\in[k]}$ are not changed, and $b'_{k-1} + b'_k = b_{k-1} + b_k$, if suffices to check condition (2) for $j = k - 1$:

$$\begin{aligned}\frac{1}{4}\sum_{i\in[k-2]}(a_i + b_i) + \frac{1}{4}(a_{k-1} + b'_{k-1}) &\le \frac{1}{4}\sum_{i\in[k-1]}(a_i + b_i) + \sum_{i\in[k-1]}(a_i \cdot h_i - \frac{a_i + b_i}{4}) \\ &= \sum_{i\in[k-1]}(a_i \cdot h_i).\end{aligned}$$

Applying the induction hypothesis on $\{a_i\}_{i\in[k-1]}, \{b_i\}_{i\in[k-2]} \cup \{b'_{k-1}\}$ and $\{h_i\}_{i\in[k-1]}$,

$$\sum_{i\in[k-1]} b_i(1 - h_i) + \phi(1 - h_{k-1}) \le \frac{1}{4}(\sum_{i\in[k-1]}(a_i + b_i) + \phi) \le \sum_{i\in[k-1]} a_ih_i.$$

If $\phi = b_k$, then immediately we have

$$\begin{aligned}\sum_{i\in[k]} b_i(1 - h_i) &\le \sum_{i\in[k-1]} b_i(1 - h_i) + b_k(1 - h_{k-1}) \\ &\le \frac{1}{4}(\sum_{i\in[k-1]}(a_i + b_i) + b_k) < \frac{1}{4}\sum_{i\in[k]}(a_i + b_i),\end{aligned}$$

as desired. Otherwise we have $\phi = \sum_{i\in[k-1]}(4a_i \cdot h_i - a_i - b_i)$, and hence we have

$$a_k \cdot h_k \ge \frac{1}{4}\sum_{i\in[k]}(a_i + b_i) - \sum_{i\in[k-1]} a_i \cdot h_i = \frac{1}{4}(a_k + b_k + \phi) = \frac{1}{4}(a_k + b'_k),$$

which implies $b'_k(1-h_k) \le \frac{1}{4}(a_k + b'_k)$. Hence we have

$$\sum_{i\in[k]} b_i(1-h_i) = \sum_{i\in[k-2]} b_i(1-h_i) + b'_{k-1}(1-h_{k-1}) + b'_k(1-h_k) + \phi(h_{k-1}-h_k)$$
$$\le \frac{1}{4}\sum_{i\in[k-2]}(a_i+b_i) + \frac{a_{k-1}+b'_{k-1}}{4} + \frac{a_k+b'_k}{4} = \frac{1}{4}\sum_{i\in[k]}(a_i+b_i),$$

which completes the induction. ◀

Given Claim 3.2, we are now ready to proof the Leftover Lemma.

Proof of Lemma 3. As before, it suffices to prove the lemma for marginal idle times, as we have $\frac{d\Delta_t}{dt} \le \frac{d\mathsf{W}_t}{dt}$ (while $\frac{d(\frac{1}{4}tm)}{dt} > 0$) for non-idle time t. Now suppose t is a marginal idle time. As before, let $\Delta_t(t)$ and $\mathsf{W}_t(t)$ be the values of variables when the algorithm is run until time t.

We prove a stronger statement that $\Delta_t(t) \le \frac{1}{4}tm + \mathsf{W}_t(t)$, by induction on the number k of marginal idle times at or before time t. Note that the stronger statement implies the lemma, as we have $\Delta_t - \Delta_t(t) = \mathsf{W}_t - \mathsf{W}_t(t)$.

In the following, we use a weaker version of Claim 3.1: we only need $\mathsf{A}_t \le \sum_{j\in J(t)} \delta_j$.

Base Case: $k = 1$. Since t is the first marginal idle time, let g be the first idle time, we know that $[g,t]$ is the only idle period. Define $J := \{j \in J(t) : s_j(t) \le g\}$ to be the set of jobs that are processed from time g to t. By definition we have $\mathsf{I}_t \le (t-g)(m-|J|)$. Recall that $\mathsf{A}_t \le \sum_{j\in J(t)} \delta_j$, where δ_j is the total pending time of job j before time t. Hence we have $\delta_j \le g$ if $j \in J$, and $\delta_j = 0$ otherwise. By Claim 3.1 we have

$$\Delta_t(t) \le \min\{\mathsf{A}_t, \mathsf{I}_t\} + \mathsf{W}_t(t) \le \min\{g|J|, (t-g)(m-|J|)\} + \mathsf{W}_t(t) \le \frac{1}{4}tm + \mathsf{W}_t(t).$$

Induction. Now suppose the statement holds for all marginal idle times $0 < t_1 < t_2 < \ldots < t_{k-1}$, and consider the next marginal idle time t_k. We show that $\Delta_{t_k}(t_k) \le \frac{1}{4}t_k m + \mathsf{W}_{t_k}(t_k)$. First of all, observe that the difference in $\Delta_{t_k}(t_k)$ and $\Delta_{t_j}(t_j)$ must come from the idle periods in $[t_j, t_k]$ and wastes created in $[t_j, t_k]$. Hence for all $j < k$ we have

$$\Delta_{t_k}(t_k) - \Delta_{t_j}(t_j) \le (\mathsf{I}_{t_k} - \mathsf{I}_{t_j}) + \mathsf{W}_{t_k}(t_k) - \mathsf{W}_{t_j}(t_j).$$

Hence, if there exists some j such that $\mathsf{I}_{t_k} - \mathsf{I}_{t_j} \le \frac{1}{4}(t_k - t_j)m$, then by induction hypothesis,

$$\Delta_{t_k}(t_k) \le \Delta_{t_j}(t_j) + \frac{1}{4}(t_k - t_j)m + \mathsf{W}_{t_k}(t_k) - \mathsf{W}_{t_j}(t_j) \le \frac{1}{4}t_k m + \mathsf{W}_{t_k}(t_k),$$

and we are done. Now suppose otherwise.

For all $i \le k$, let $g_i \in (t_{i-1}, t_i)$ be the first idle time after t_{i-1} (assume $g_0 = t_0 = 0$), i.e., $[g_1,t_1],[g_2,t_2],\ldots,[g_k,t_k]$ are the disjoint idle periods. Define $J_i := \{j \in J(t_k) : r_j \in [t_{i-1}, g_i)\}$. Note that for all $j \in J_i$, we have $\delta_j \le \sum_{x=i}^{k}(g_x - t_{x-1})$, as j is not pending during idle periods; for $j \in J(r_k) \setminus \cup_{i\le k} J_i$, we have $\delta_i = 0$. For all $i \in [k]$, define

$$a_i := t_{k-i+1} - g_{k-i+1}, b_i := g_{k-i+1} - t_{k-i}, h_i := 1 - \tfrac{1}{m}\textstyle\sum_{x\in[k-i+1]}|J_x|.$$

We show that the three sequences of positive reals $\{a_i\}_{i\in[k]}$, $\{b_i\}_{i\in[k]}$, $\{h_i\}_{i\in[k]}$ satisfy the conditions of Claim 3.2, which implies $\Delta_{t_k}(t_k) \le \mathsf{A}_{t_k} + \mathsf{W}_{t_k}(t_k) \le \frac{1}{4}t_k m + \mathsf{W}_{t_k}(t_k)$ as

$$\begin{aligned}\mathsf{A}_{t_k} &\le \sum_{j\in J(r_k)} \delta_j \le \sum_{i\in[k]} \left(\sum_{x=i}^{k}(g_x - t_{x-1})\right)|J_i| = \sum_{i\in[k]} \left((g_i - t_{i-1}) \cdot \sum_{x\in[i]} |J_x|\right) \\ &= m\sum_{i\in[k]}(b_{k-i+1}\cdot(1-h_{k-i+1})) \le \frac{1}{4}m\sum_{i\in[k]}(a_i+b_i) = \frac{1}{4}t_k m.\end{aligned}$$

Finally, we check the conditions of Claim 3.2. Condition (1) trivially holds. For condition (2), observe that $\mathsf{I}_{t_i} - \mathsf{I}_{t_{i-1}} \le (t_i - g_i)\cdot(m - \sum_{x\in[i]}|J_x|) = a_{k-i+1}\cdot h_{k-i+1}\cdot m$. Hence we have

$$\sum_{i\in[j]}(a_i\cdot h_i) \ge \frac{1}{m}\sum_{i\in[j]}(\mathsf{I}_{t_{k-i+1}} - \mathsf{I}_{k-i}) = \frac{1}{m}(\mathsf{I}_{t_k} - \mathsf{I}_{t_{k-j}}) > \frac{1}{4}(t_k - t_{k-j}) = \frac{1}{4}\sum_{i\in[j]}(a_i+b_i),$$

as required. ◀

4 Breaking 1.5 on Identical Machines

In this section, we prove Theorem 1. We will prove by contradiction: assume for contrary that $\mathsf{ALG} > 1+\gamma$, we seek to derive a contradiction, e.g., no schedule could complete all jobs before time 1 (Recall that we assume $\mathsf{OPT} = 1$). To do so, we introduce two types of arguments: we use a bin-packing argument to show that the last job must be of small size, as otherwise there exists a set of infeasible large jobs; then we use an efficiency argument (built on the Leftover Lemma) to show that the total processing (excluding idle and waste periods) our algorithm completes exceed m, the maximum possible processing OPT does.

For convenience of presentation, in the rest of the paper, we adopt the *minimum counter-example* assumption [25], i.e., we consider the instance with the minimum number of jobs such that $\mathsf{ALG} > 1+\gamma$ and $\mathsf{OPT} = 1$. As an immediate consequence of the assumption, we get that no job arrives after s_n. This is because such jobs do not affect the start time of n and therefore could be removed to obtain a smaller counter example.

Recall that in our algorithm, we set $\alpha = \frac{1}{200}$ and $\beta = \sqrt{2}-1$. Define $\gamma := \frac{1}{2} - \epsilon$, where $\epsilon = \frac{1}{20000}$. We first provide some additional structural properties of our algorithm, which will be the building blocks of our later analysis.

4.1 Structural Properties

Observe that if a job is replaced, then it must be the minimum job among the m jobs that are currently being processed. Hence immediately we have the following lemma, since otherwise we can find $m+1$ jobs (including the replacer) of size larger than $\frac{1}{2}$.

▶ **Fact 4.1** (Irreplaceable Jobs). *Any job with size at least $\frac{1}{2}$ cannot be replaced.*

Next, we show that if a job i is pending for a long time, then each of the jobs processed at time s_i must be of (relatively) large size.

▶ **Lemma 8.** *For any job i, we have $p_j > \min\{s_i - r_i, p_i\}$ for all $j \in J(s_i)$.*

Proof. It suffices to consider the non-trivial case when $s_i > r_i$. Consider any $j \in J(s_i)$. If $p_j > p_i$ or $s_j(s_i) < r_i$[4], then we have $p_j > \min\{s_i - r_i, p_i\}$. Otherwise, we consider time

[4] Note that we use $s_j(s_i)$ here instead of s_j as j can possibly be replaced after time s_i.

$s_j(s_i)$, at which job j is scheduled. Since $p_j < p_i$ and $s_j(s_i) > r_i$ (i is already released), we know that p_i must be processed at $s_j(s_i)$. Hence we know that i is replaced during $(s_j(s_i), s_i)$, which is impossible since j (which is of smaller size than i) is being processed during this period. ◀

Specifically, since $\mathsf{ALG} = s_n + p_n > 1 + \gamma$ and $r_n + p_n \leq \mathsf{OPT} = 1$, we have $s_n - r_n > \gamma$. Applying Lemma 8 to job n gives the following.

▶ **Corollary 9** (Jobs Processed at Time s_n). *We have $p_j > \min\{\gamma, p_n\}$ for all $j \in J(s_n)$.*

In the following, we show two lemmas, one showing that if a job released very early is not scheduled, then all jobs processed at that time are (relatively) large; the other showing that if a job is replaced, then the next time it is scheduled must be the completion time of a larger job.

▶ **Lemma 10** (Irreplaceable Jobs at Arrival). *If a job k is not scheduled at r_k and $r_k < \alpha p_k$, then $p_j \geq \frac{p_k}{1+\beta}$ for all $j \in J(r_k)$.*

Proof. By our replacement rule, k is not scheduled at r_k either because k is not the largest pending job at r_k, or k is the largest pending job, but the minimum job in $J(r_k)$ is not replaceable.

For the second case, since the minimum job i in $J(r_k)$ is processed at most $r_k < \alpha p_k$, job i must violate our third replacement rule, that is $p_i \geq \frac{p_k}{1+\beta}$. For the first case, let k' be the first job of size at least p_k that is not scheduled at its release time. Then we have $r_{k'} < r_k < \alpha p_k < \alpha p_{k'}$. Hence by the above argument every job in $J(r_{k'})$ is of size at least $\frac{p_k}{1+\beta} > r_k$. Thus every job in $J(r_k)$ is also of size at least $\frac{p_k}{1+\beta}$. ◀

▶ **Lemma 11** (Reschedule Rule). *Suppose some job k is replaced, then the next time k is scheduled must be the completion time of a job j such that $p_k < p_j \leq s_k$.*

Proof. Suppose k is replaced at time t and rescheduled at time t'. Since k can only replace other jobs at r_k, the next time k is scheduled must be when some machine becomes idle. And this happens only if the job j processed before t' on this machine is completed. Moreover, since k is pending from t to t', if $s_j \geq t$, i.e., k is pending when j is scheduled, then (by greedy scheduling rule) we have $p_j > p_k$; otherwise j is being processed at time t, and we also have $p_j > p_k$ as k is the smallest job among all jobs in $J(t)$ by the replacement rule. Hence, we have $s_k \geq c_j \geq p_j > p_k$. ◀

We present the central lemma for our bin-packing argument as follows. Intuitively, our bin-packing argument applies if there exists time t_1 and t_2 that are far apart, and the jobs in $J(t_1)$ and $J(t_2)$ are large (e.g. larger than $\frac{1}{3}$): if $J(t_1) \cap J(t_2) = \emptyset$, then together with job n, we have found an infeasible set of $2m + 1$ large jobs; otherwise (since t_1 and t_2 are far apart) we show that the jobs in $J(t_1) \cap J(t_2)$ must be even larger, e.g. larger than $\frac{2}{3}$.

▶ **Lemma 12** (Bin-Packing Constraints). *Given non-idle times t_1, t_2 such that $t_1 < t_2$ and $n \notin J(t_1) \cup J(t_2)$, let $a = \min_{j \in J(t_1)}\{p_j\}$ and $b = \min_{j \in J(t_2)}\{p_j\}$, none of the following cases can happen:*
(1) *$a > \frac{1}{3}$, $b > \frac{2}{3(1+\beta)}$, $t_2 - t_1 > \frac{2}{3}$ and $p_n > \frac{1}{3}$;*
(2) *$\min\{a, b, p_n\} > \frac{1}{2+\beta}$, and $t_2 - t_1 > 1 - \min\{a, b, p_n\}$.*

Proof. We show that if any of the cases happens, then we have the contradiction that $\mathsf{OPT} > 1$. We first consider case (1). We show that we can associate jobs to machines such that every machine is associated with either a job of size larger than $\frac{2}{3}$, or two jobs of size

larger than $\frac{1}{3}$. Moreover, we show that every job is associated once, while n is not associated. Note that since $p_n > \frac{1}{3}$, such an association would imply the contradiction that $\mathsf{OPT} > 1$.

First, we associate every $j \in J(t_2)$ to the machine that it is processed on. If $p_j > \frac{2}{3}$ then we are done with this machine; otherwise (when $p_j \le \frac{2}{3}$), we have $s_j(t_2) > t_1$ and we show that we can associate another job of size larger than $\frac{1}{3}$ to this machine.

- If the job $i \in J(t_1)$ processed on this machine is not replaced, or $p_i \le \frac{2}{3(1+\beta)}$, then we associate i with this machine (it is easy to check that i has not been associated before);
- otherwise the first job that completes after t_1 must be of size larger than $(1+\beta)p_i > \frac{2}{3}$, which also has not been associated before. Thus we associate it to this machine.

In both cases we are able to do the association, as claimed.

Next we consider case (2). Consider any job $i \in J(t_1)$, we have $p_i > \frac{1}{2+\beta}$. We apply an association argument similar as before: let M be the machine that job i is processed on.

- If i is replaced, then we associate the first job that completes on this machine after i is replaced, which is of size larger than $\frac{1+\beta}{2+\beta} > 1 - \min\{a, b, p_n\}$, to machine M;
- otherwise if $p_i > 1 - \min\{a, b, p_n\}$ then we associate i to M;
- otherwise we know that i completes before t_2, and we can further associate to M the job in $J(t_2)$ processed on M.

It is easy to check that every job is associated at most once. Hence every machine is associated with either a job of size larger than $1 - \min\{a, b, p_n\}$, or two jobs of size larger than $\min\{a, b, p_n\}$, which (together with $p_n > \frac{1}{2+\beta}$) gives $\mathsf{OPT} > 1$, a contradiction. ◀

4.2 Upper Bounding p_n: Bin-Packing Argument

We show in this section how to apply the structural properties from Section 4.1 to provide an upper bound $\frac{1}{2+\alpha}$ on p_n. Recall that we assume $\mathsf{ALG} > 1 + \gamma$. We show that if $p_n > \frac{1}{2+\alpha}$, then Lemma 12 leads us to a contradiction. We first prove the following lemma (which will be further used in Section 4.4), under a weaker assumption, i.e., $p_n > \frac{1}{2+\beta}$.

▶ **Lemma 13.** *If $p_n > \frac{1}{2+\beta}$, then job n is never replaced.*

Proof. Assume the contrary and consider the last time when n is replaced. Note that by Fact 4.1, we have $p_n < \frac{1}{2}$. Suppose n is replaced by job l at time r_l. Then by our replacement rule, we have $p_l \ge (1+\beta)p_n$. As $r_l + p_l \le 1$ and $s_n + p_n > 1 + \gamma$, we have

$$s_n - r_l > (1 + \gamma - p_n) - (1 - p_l) > \gamma + \beta p_n > 1 - p_n > p_n,$$

where the second last inequality holds since $\gamma > \frac{1}{2+\beta}$ and $p_n > \frac{1}{2+\beta}$. Since $r_n < r_l$, by Lemma 8, we have $\min_{j \in J(s_n)}\{p_j\} > \min\{s_n - r_n, p_n\} = p_n$. Thus we can apply Lemma 12(2), with $t_1 = r_l$, $t_2 = s_n$, $a = \min_{j \in J(t_1)}\{p_j\} = p_n$ and $b = \min_{j \in J(t_2)}\{p_j\} > p_n$, and derive a contradiction. ◀

▶ **Lemma 14** (Upper Bound on Last Job). *We have $p_n \le \frac{1}{2+\alpha}$.*

For space reasons, we defer its proof to Appendix B.

Given the upper bound $\frac{1}{2+\alpha}$ on p_n, we show the following stronger version of Lemma 13 that any job of size larger than $\frac{1}{2+\beta}$ cannot be replaced.

▶ **Corollary 15** (Irreplaceable Threshold). *Any job of size larger than $\frac{1}{2+\beta}$ cannot be replaced.*

Proof. Assume the contrary that some job j of size larger than $\frac{1}{2+\beta}$ is replaced (say, by job k at r_k). Then we have $p_k > (1+\beta)p_j$ and $\min_{i\in J(r_k)}\{p_i\} = p_j > \frac{1}{2+\beta}$. On the other hand, by Corollary 9, we have $\min_{i\in J(s_n)}\{p_i\} > \min\{p_n, \gamma\} > \frac{1}{2+\beta}$. Since $p_n < \frac{1}{2+\alpha} < \gamma$, we have

$$s_n - r_k > (1+\gamma-p_n) - (1-\frac{1+\beta}{2+\beta}) = \gamma - p_n + \frac{1+\beta}{2+\beta} > \frac{1+\beta}{2+\beta}.$$

Hence we can apply the bin-packing argument to derive a contradiction: by Lemma 12(2), with $t_1 = r_k$, $t_2 = s_n$, $a = p_j$ and $b > \frac{1}{2+\beta}$, we have a contradiction. ◀

4.3 Lower Bounding p_n: Efficiency Argument

Next we establish a lower bound on p_n, applying the Leftover Lemma. First, observe that if n is never replaced, then n is pending from r_n to s_n, where $r_n \le 1 - p_n$; if n is replaced, then n is pending from r_l to s_n, where r_l is the last time n is replaced. Since $r_l \le 1 - (1+\beta)\cdot p_n < 1 - p_n$, in both cases n is pending during time period $[1-p_n, s_n)$. Hence we have the following fact.

▶ **Fact 4.2** (Non-idle Period). *Every $t \in [1-p_n, s_n)$ is non-idle.*

As a warm-up, we show the following simple lower bound on p_n using the Leftover Lemma.

▶ **Lemma 16** (Simple Lower Bound). *We have $p_n > \frac{1}{3} - 2\alpha$.*

Proof. Let $t = 1 - p_n$. Since there is no waste after time 1, the total waste located after time t is $\mathsf{W}_1 - \mathsf{W}_t$. Since no machine is idle during time $[1-p_n, s_n)$, the total processing our algorithm does after time t is at least $m(s_n - t) - (\mathsf{W}_1 - \mathsf{W}_t)$. On the other hand, the total processing our algorithm does after time t is upper bounded by $m(1-t) + \Delta_t$ (Observation 3.1). Applying Lemma 3 (the Leftover Lemma) on Δ_t, we have

$$m(s_n - t) < m(1-t) + (\mathsf{W}_1 - \mathsf{W}_t) + \Delta_t \le m(1-t) + \mathsf{W}_1 + \frac{1}{4}tm \le m(1-t) + \alpha m + \frac{1}{4}tm.$$

Note that the last inequality follows since (by our replacement rule) each job j can only create a waste of size at most $\alpha \cdot p_j$, and the total size of jobs is at most m. Thus we have

$$\mathsf{ALG} = s_n + p_n \le 1 + \alpha + \tfrac{1-p_n}{4} + p_n = \tfrac{5}{4} + \alpha + \tfrac{3}{4}p_n,$$

which implies that (recall that we assume $\mathsf{ALG} > 1 + \gamma = \frac{3}{2} - \epsilon$)

$$p_n \ge \tfrac{4}{3}\cdot(\mathsf{ALG} - \tfrac{5}{4} - \alpha) > \tfrac{4}{3}\cdot(\tfrac{3}{2} - \epsilon - \tfrac{5}{4} - \alpha) = \tfrac{1}{3} - \tfrac{4}{3}(\epsilon + \alpha) > \tfrac{1}{3} - 2\alpha,$$

where the last inequality holds by our choices of parameter, i.e., $\epsilon = \frac{1}{20000}$ and $\alpha = \frac{1}{200}$. ◀

Observe that the Leftover Lemma provides tighter upper bounds for smaller values of t. Thus in the above proof, if we can find a smaller t such that there is no (or very little) idle time from t to s_n, then we can obtain a stronger lower bound on p_n.

▶ **Lemma 17** (Lower Bound on Last Job). *We have $p_n > \frac{1}{2+\beta}$.*

Proof. Assume for contrary that $p_n \le \frac{1}{2+\beta}$. Then by Corollary 9, we have $p_j > \min\{\gamma, p_n\} = p_n$ for all job $j \in J(s_n)$. Hence at least two jobs $k, j \in J(s_n) \cup \{n\}$ are scheduled on the same machine in OPT, such that $r_k < 1 - 2p_n$. We prove the following claim[5], which enables us to use a refined efficiency argument, i.e., apply the Leftover Lemma on a earlier time $t = r_k$. For space reasons, we omit its proof.

[5] We remark that the proof of Claim 4.1 relies on the fact that p_n is not too small. Hence Lemma 16 (the warm-up lower bound) is necessary for achieving the improved lower bound (Lemma 17).

▶ **Claim 4.1.** *The total idle time* $\mathsf{I}_{s_n} - \mathsf{I}_{r_k}$ *during time period* $[r_k, s_n]$ *is at most* $3\alpha \cdot m$.

By the above claim, the total processing ALG does after time $t = r_k$ is at least $m(s_n - t) - (\mathsf{W}_1 - \mathsf{W}_t) - 3\alpha m$. On the other hand, by Observation 3.1 and Lemma 3 we have

$$m(s_n - t) - (\mathsf{W}_1 - \mathsf{W}_t) - 3\alpha m \le m(1-t) + \Delta_t \le m(1-t) + \tfrac{1}{4}tm + \mathsf{W}_t.$$

Rearranging the inequality, we have (recall that $t = r_k < 1 - 2p_n$)

$$1 + \gamma < \mathsf{ALG} = s_n + p_n \le 1 + \tfrac{1-2p_n}{4} + 4\alpha + p_n \le \tfrac{5}{4} + 4\alpha + \tfrac{\beta}{2(2+\beta)},$$

which is a contradiction by our choice of parameters. ◀

4.4 A Hybrid Argument

We have shown that assuming $\mathsf{ALG} > 1 + \gamma$, the size of the last job n can be bounded as $\frac{1}{2+\beta} < p_n \le \frac{1}{2+\alpha}$. In the remaining part of this section, we use a hybrid argument to show that we can either use the bin-packing argument to find a set of infeasible large jobs; or derive a contradiction using the efficiency argument.

General Framework. Given that $\frac{1}{2+\beta} < p_n \le \frac{1}{2+\alpha} < \gamma$, we have $s_n = \mathsf{ALG} - p_n > 1$. Thus we have $s_j \ne r_j$ for all $j \in J(s_n)$ (as they are processed at time $s_n > 1$). Moreover, by Corollary 9, we have $\min_{j \in J(s_n)}\{p_j\} > \min\{\gamma, p_n\} > \frac{1}{2+\beta}$. In other words, there exists a set $J(s_n) \cup \{n\}$ of large jobs, each of which is never replaced (by Corollary 15), and none of them is scheduled at its release time. We know that at least two of them, say k and j (assume k is released earlier), are scheduled on the same machine in OPT. Since $s_k - r_k > (1 + \gamma - p_n - p_k) - (1 - p_k - p_j) \ge \gamma$, we know that k is pending from r_k to s_k, which is a period of length γ. Thus either our algorithm finishes a lot of processing during this period (then we can use the efficiency argument), or there are many idle and waste periods during this period (then we can use the bin-packing argument to find another m large jobs, e.g., larger than $\frac{1}{3}$) .

For the compete analysis, please refer to the full version of our paper.

References

1. Susanne Albers. Better bounds for online scheduling. *SIAM Journal on Computing*, 29(2):459–473, 1999.
2. Nir Avrahami and Yossi Azar. Minimizing total flow time and total completion time with immediate dispatching. *Algorithmica*, 47(3):253–268, 2007.
3. Y. Bartal, A. Fiat, H. Karloff, and R. Vohra. New algorithms for an ancient scheduling problem. *Journal of Computer and System Sciences*, 51(3):359–366, 1995.
4. Piotr Berman, Moses Charikar, and Marek Karpinski. On-line load balancing for related machines. *Journal of Algorithms*, 35(1):108–121, 2000.
5. Bo Chen, André van Vliet, and Gerhard J. Woeginger. A lower bound for randomized on-line scheduling algorithms. *Information Processing Letters*, 51(5):219–222, 1994.
6. Bo Chen, André van Vliet, and Gerhard J. Woeginger. An optimal algorithm for preemptive on-line scheduling. *Operations Research Letters*, 18(3):127–131, 1995.
7. Bo Chen, André van Vliet, and Gerhard J. Woeginger. New lower and upper bounds for on-line scheduling. *Operations Research Letters*, 16(4):221–230, 1994.
8. Bo Chen and Arjen P. A. Vestjens. Scheduling on identical machines: How good is LPT in an on-line setting? *Operations Research Letters*, 21(4):165–169, 1997.

9 Marek Chrobak, Wojciech Jawor, Jirí Sgall, and Tomás Tichý. Online scheduling of equal-length jobs: Randomization and restarts help. *SIAM Journal on Computing*, 36(6):1709–1728, 2007.

10 György Dósa and Leah Epstein. Online scheduling with a buffer on related machines. *Journal of Combinatorial Optimization*, 20(2):161–179, 2010.

11 György Dósa and Leah Epstein. Preemptive online scheduling with reordering. *SIAM Journal on Discrete Mathematics*, 25(1):21–49, 2011.

12 Tomás Ebenlendr, Wojciech Jawor, and Jirí Sgall. Preemptive online scheduling: Optimal algorithms for all speeds. *Algorithmica*, 53(4):504–522, 2009.

13 Matthias Englert, Deniz Özmen, and Matthias Westermann. The power of reordering for online minimum makespan scheduling. *SIAM Journal on Computing*, 43(3):1220–1237, 2014.

14 Leah Epstein, John Noga, Steven S. Seiden, Jirí Sgall, and Gerhard J. Woeginger. Randomized online scheduling on two uniform machines. In *SODA*, pages 317–326. ACM/SIAM, 1999.

15 Leah Epstein and Jirí Sgall. A lower bound for on-line scheduling on uniformly related machines. *Operations Research Letters*, 26(1):17–22, 2000.

16 Rudolf Fleischer and Michaela Wahl. On-line scheduling revisited. *Journal of Scheduling*, 3(6):343–353, 2000.

17 Ronald L. Graham. Bounds on multiprocessing timing anomalies. *SIAM Journal of Applied Mathematics*, 17(2):416–429, 1969.

18 Han Hoogeveen, Chris N Potts, and Gerhard J Woeginger. On-line scheduling on a single machine: maximizing the number of early jobs. *Operations Research Letters*, 27(5):193–197, 2000.

19 Zhiyi Huang, Ning Kang, Zhihao Gavin Tang, Xiaowei Wu, and Yuhao Zhang. Online makespan minimization: The power of restart. *CoRR (to appear in APPROX 2018)*, abs/1806.02207, 2018.

20 J. F. Rudin III. *Improved Bound for the Online Scheduling Problem*. PhD thesis, University of Texas at Dallas, 2001.

21 J. F. Rudin III and R. Chandrasekaran. Improved bounds for the online scheduling problem. *SIAM Journal on Computing*, 32(3):717–735, 2003. `arXiv:http://dx.doi.org/10.1137/S0097539702403438`.

22 David R. Karger, Steven J. Phillips, and Eric Torng. A better algorithm for an ancient scheduling problem. *Journal of Algorithms*, 20(2):400–430, 1996.

23 Hans Kellerer, Vladimir Kotov, Maria Grazia Speranza, and Zsolt Tuza. Semi on-line algorithms for the partition problem. *Operations Research Letters*, 21(5):235–242, 1997.

24 Shisheng Li, Yinghua Zhou, Guangzhong Sun, and Guoliang Chen. Study on parallel machine scheduling problem with buffer. In *IMSCCS*, pages 278–273. IEEE Computer Society, 2007.

25 John Noga and Steven S. Seiden. An optimal online algorithm for scheduling two machines with release times. *Theoretical Computer Science*, 268(1):133–143, 2001.

26 Jianjun Wen and Donglei Du. Preemptive on-line scheduling for two uniform processors. *Operations Research Letters*, 23(3-5):113–116, 1998.

27 Guochuan Zhang. A simple semi on-line algorithm for p2//c_{max} with a buffer. *Information Processing Letters*, 61(3):145–148, 1997.

A Overview of Techniques for the Two Machine Case

Our analysis for the two-machine case follows the same framework as the general case: we use a bin-packing argument to upper bound the size of last job, and use an efficiency argument to lower bound the size of last job, and finally use a hybrid argument to handle the boundary case. In this section, we will overview two technical ingredients that are specifically developed for the two-machine case, namely, a structural result that upper bounds the number of times that large jobs are replaced, and a refined efficiency argument. Similar to the general case, most of the difficulties arise when the last job has medium size. For concreteness, readers may consider the size of the last job n as slightly larger than $\gamma = 0.38$, say, $p_n = 0.4$.

Recall that for the two-machine case we fix the parameters $\beta = \alpha = 0.2$.

A.1 Bounding Major Replacements

We are interested in jobs that have size at least p_n. We call such jobs *major jobs* and refer to the replacements of major jobs as *major replacements*. In the case that there are only two machines, the number of major jobs is at most 4 by a simple bin-packing argument. (Recall that we focus on medium size last job, say $p_n = 0.4$.) Further, only major jobs can replace major jobs. Hence, we can show sharp bound on the number of major replacements. In particular, we show that when the last job is of medium size, there is either no major replacement, or at most one major replacement, depending on the size of the last job. This is formulated as the following lemmas.

▶ **Lemma 18.** *If ALG* > 1.38 *and* $p_n > \frac{1}{2+\alpha}$*, then there is no major replacement.*

Proof. Recall that for the two machines case, we set $\beta = \alpha$. First we show that job n cannot be replaced given $p_n > \frac{1}{2+\alpha}$. Observe that the replacer of n must be of size $(1+\alpha)p_n > 1 - \frac{1}{2+\alpha}$, which cannot be scheduled on the same machine with n in the optimal schedule. Suppose n is replaced (which can happen at most once) by l. Then we know that n is pending from r_l to s_n, where

$$s_n - r_l > (1.38 - p_n) - (1 - (1+\alpha)p_n) = 0.38 + \alpha p_n > \frac{1}{2+\alpha}.$$

Hence by Lemma 8, we have $p_k, p_j > \frac{1}{2+\alpha}$, where $\{k, j\} = J(s_n)$ are the two jobs processed at time s_n. If $l \notin \{k, j\}$, then none of k, j, n can be scheduled on the same machine with l in OPT, which yields a contradiction; otherwise suppose $l = k$. Then we must have $J(r_l) = \{n, j\}$ as otherwise we also have the same contradiction as just argued. Then n and j must be scheduled on the same machine in OPT. If $r_n < r_j$, then $r_n < 1 - p_n - p_j < 1 - \frac{2}{2+\alpha} < \alpha p_n$. Hence n must be scheduled at r_n, as otherwise by Lemma 10 the two jobs in $J(r_n)$ are of size larger than $\frac{p_n}{1+\alpha} > \frac{1}{3}$. Since

$$r_l - r_n > s_j - r_n > (1.38 - p_n - p_j) - (1 - p_n - p_j) = 0.38 > \alpha,$$

it is impossible for l to replace n as n has been processed a portion larger than α. If $r_j < r_n$, then for the same reasoning j is scheduled at r_j. As $s_j - r_j > (1.38 - p_n - p_j) - (1 - p_n - p_j) = 0.38$, we know that j is replaced, which is impossible as by Lemma 11, the job completed at s_j is a job of size larger than $\frac{1}{2+\alpha}$, apart from l, j and n.

Hence we know that n is never replaced. Now suppose some other job of size larger than $\frac{1}{2+\alpha}$ is replaced at time r_x. Then we have $p_x > 1 - \frac{1}{2+\alpha}$, while the two jobs in $J(r_x)$ are of size larger than $\frac{1}{2+\alpha}$. Note that since n is never replaced, it is not scheduled before s_n. Further by our assumption that no job arrives after s_n, we have $r_x < s_n$. Thus $n \notin J(r_x)$, which implies a contradiction. ◀

▶ **Lemma 19.** *If ALG* > 1.38 *and* $p_n \in (0.38, \frac{1}{2+\alpha}]$*, then there is at most one major replacement.*

The main idea is that, when there are two or more major replacements, then we can find at least four major jobs. Hence as long as we can find a job of size "not too small", then the bin-packing argument yields a contradiction.

Why is bounding the number of major replacements useful? Recall that in the efficiency argument, we upper bound the total waste by $\alpha m = 2\alpha$ using the fact that each job can only create waste once at its release time if it replaces some other job, and the amount of waste created is at most α times the size of the replacer. Suppose we can find one major job that does not replace any other job at its arrival. Then, the upper bound of the total waste will significantly decrease by αp_n. (Recall that we focus on medium size last job, say $p_n = 0.4$.)

How do we find major jobs that do not replace other jobs at their arrival? It turns out we can argue that in order to have $\mathsf{ALG} > 1 + \gamma$, there must exists some major jobs whose final start times do not equal their release times. For example, the last job n's final start time definitely does not equal its release time. For each of these jobs, if it replaces some other job at its arrival, it must be replaced later on in order to get rescheduled at its final start time. Hence, a major replacement must occur for each of these jobs. By bounding the number of major replacements, we get that some of these major jobs must not replace other jobs at their arrivals.

A.2 Refined Efficiency Argument

The second technical ingredient is a more careful efficiency argument. Let t be the last idle time before s_n. The total amount of work done in the optimal schedule after time t is at most $2(1-t)$. (Recall that $\mathsf{OPT} = 1$ and there are $m = 2$ machines.)

How much work does the algorithm process after time *t*? The algorithm is fully occupied from time t to time s_n. Further, let P be the total processing our algorithm does after s_n. Then, the total amount of processing power after time t by the algorithm is $2(s_n - t) + \mathsf{P}$. However, some of the processing power is wasted (due to replacements) and some of the workload could have been done before time t in OPT (the leftover). Recall that W_t is the amount of waste before time t. Hence, the amount of waste located after time t is $\mathsf{W}_1 - \mathsf{W}_t$. Also recall that the amount of leftover is denoted as Δ_t. So we have: (the first inequality comes from Observation 3.1)

$$2(1-t) \geq 2(s_n - t) + \mathsf{P} - \Delta_t - (\mathsf{W}_1 - \mathsf{W}_t) \geq 2(s_n - t) + \mathsf{P} - \frac{t}{2} - \mathsf{W}_1,$$

where the second inequality follows by the leftover lemma. Rearranging terms, the above implies:

$$\mathsf{ALG} = s_n + p_n \leq 1 + p_n - \frac{\mathsf{P}}{2} + \frac{t}{4} + \frac{\mathsf{W}_1}{2}.$$

Then, we will bound each of the terms P, t, and W_1: P is trivially lower bounded by p_n; t is trivially upper bounded by 1, while a more careful argument shows that $t \leq 1 - p_n$ (Fact 4.2); and W_1 is trivially upper bounded by 2α. If we plug in the trivial bounds, we get: $\mathsf{ALG} \leq 1.25 + \alpha + \frac{p_n}{2}$.

Hence, we recover the efficiency argument similar to what we have used in Section 4.1 (except that we further relax $\frac{m-1}{m} p_n \leq p_n$ in the general case). However, if we can obtain

improved bounds on P, t, or W_1, we would have a better efficiency argument for upper bounding ALG. It is usually impossible to get better bounds for all of these three quantities. Nonetheless, we manage to do so for at least one of them in all cases.

We have already provided an argument for getting a better upper bound of W_1 by bounding the number of major replacements. Next, we present some intuitions why it is possible to get improved bounds for P and t.

Since the jobs are scheduled greedily, if a replaced job x is of size smaller than p_n, then it often happens that job x is rescheduled after s_n. Hence while we suffer a loss in the total waste W_1 due to the waste that comes from x, we have an extra gain of p_x in P. In general, we will develop a unified upper bound on $\mathsf{W}_1 - \mathsf{P}$, which measures the net waste due to replacements.

Apart from the trivial upper bound $1 - p_n$ on t (by Fact 4.2), we can often derive better upper bounds on t if we do not have a good upper bound on $\mathsf{W}_1 - \mathsf{P}$. In the case when the total net waste is large, we can usually find many major jobs processed at the end of the schedule. As we have only $m = 2$ machines, during idle periods, only one job can be processed while no job is pending. Suppose we find four major jobs processed at the end of the schedule, then at least three of them must be released after the last idle time t. Since $\mathsf{OPT} = 1$, we must have $t \leq 1 - 2p_n$, which gives a better upper bound on t.

B Missing Proofs

Proof of Lemma 14. We first show a weaker upper bound: $p_n \leq \frac{1+\beta}{2}$. Assume the contrary that $p_n > \frac{1+\beta}{2} > \frac{1}{2}$. As shown in the proof of Lemma 8, for all $j \in J(s_n)$, if $s_j > r_n$, we have $p_j > p_n > \frac{1+\beta}{2} > 1-\gamma$; otherwise $s_j < r_n$ and we have $p_j > s_n - r_n \geq (1+\gamma-p_n)-(1-p_n) = \gamma$. Among the $m+1$ jobs $J(s_n)\cup\{n\}$, there exist two jobs, say k and j, that are scheduled on the same machine in OPT. Moreover, we have $s_k, s_j < r_n$, since otherwise one of them is larger than $1-\gamma$ and they cannot be completed in the same machine within makespan 1. Let k be the one with a smaller release time, i.e., $r_k < r_j$. Then we have $r_k \leq 1 - p_k - p_j < 1 - 2\gamma < \alpha p_k$.

Observe that k is never replaced, as otherwise (by Lemma 11, the reschedule rule) we have $s_k > p_k$, which implies $r_n + p_n > s_k + p_n > p_k + p_n > \gamma + \frac{1+\beta}{2} > 1$, a contradiction.

By Lemma 10, we know that $s_k = r_k$ (k is scheduled at its arrival time r_k), as otherwise the minimum job in $J(r_k)$ is of size at least $\frac{\gamma}{1+\beta}$. Thus we have $s_k \geq \frac{\gamma}{1+\beta}$, which is also a contradiction as $r_n + p_n > s_k + p_n > \frac{\gamma}{1+\beta} + \frac{1+\beta}{2} > 1$ (recall that $\gamma = \frac{1}{2} - \epsilon$ and $\beta = \sqrt{2} - 1$).

By $r_k + p_k + p_j \leq 1$ and $r_k + p_k + p_n > 1 + \gamma$, we have $p_n > 2\gamma$. Hence $r_n < 1 - 2\gamma < \alpha p_n$. By Lemma 10, we know that the minimum job in $J(r_n)$ is of size at least $\frac{p_n}{1+\beta} > \frac{1}{2}$. Hence we have the contradiction that there are $m+1$ jobs, namely $J(r_n)\cup\{n\}$, of size larger than $\frac{1}{2}$.

Hence we have that $p_n \leq \frac{1+\beta}{2}$. Now assume that $p_n > \frac{1}{2+\alpha}$.

By Lemma 13, we know that n is never replaced. By Corollary 9, all jobs in $J(s_n)\cup\{n\}$ are of size at least $\min\{\gamma, p_n\} \geq \frac{1}{2+\alpha}$. Let $k, j \in J(s_n)\cup\{n\}$ be scheduled on the same machine in OPT such that $r_k \leq 1 - p_k - p_j < 1 - \frac{2}{2+\alpha} \leq \alpha p_k$. Note that different from the previous analysis (when $p_n > \frac{1+\beta}{2}$), it is possible that $n \in \{k, j\}$.

By Lemma 10, if k is not scheduled at r_k, then each job $i \in J(r_k)$ has size $p_i \geq \frac{p_k}{1+\beta} \geq \frac{1}{(2+\alpha)\cdot(1+\beta)} > \frac{1}{3}$. Then we can apply Lemma 12(1) with $t_1 = r_k$ and $t_2 = s_n$ to derive a contradiction (observe that $t_2 - t_1 = s_n - r_k > (1+\gamma-p_n) - (1-p_k-p_j) > \frac{3}{2+\alpha} - \frac{1+\beta}{2} \geq \frac{2}{3}$).

Hence we know that k is scheduled at time r_k (thus we conclude that $k \neq n$).

We show that k must be replaced (say, by job l at time r_l) , as otherwise by $r_k + p_k + p_n > 1 + \gamma$ and $r_k + p_k + p_j \leq 1$, we have $p_n > \gamma + p_j > \gamma + \frac{1}{2+\alpha} > \frac{1+\beta}{2}$, which is a contradiction.

By Lemma 11, we have $s_k > p_k \geq \frac{1}{2+\alpha}$. We also have that $p_n \leq 1 - \frac{1}{2+\alpha}$, as otherwise $r_n \leq 1 - p_n < \frac{1}{2+\alpha} < s_k$ (k is scheduled at s_k, when n is pending), which implies $p_k > p_n > 1 - \frac{1}{2+\alpha} > \frac{1}{2}$, contradicting Fact 4.1 (jobs larger than $\frac{1}{2}$ cannot be replaced). Since $p_l > (1+\beta)p_k$, we have

$$s_n - r_l > (1+\gamma-p_n) - (1-(1+\beta)p_k) = (1+\beta)p_k + \gamma - p_n > \frac{1+\beta}{2+\alpha} + \gamma - \frac{1+\alpha}{2+\alpha} > \frac{1+\alpha}{2+\alpha}.$$

Then we can apply Lemma 12(2), with $t_1 = r_l$, $t_2 = s_n$, $a = p_k$, $b \geq \frac{1}{2+\alpha} > \frac{1}{2+\beta}$, to derive a contradiction. ◀

C Other Candidate Algorithms

All the candidate algorithms are based on LPT. That is, whenever there is an idle machine, we always schedule the largest job. The only difference is the replacement rule. We will show that none of the them can beat the ratio of 1.5.

Candidate Algorithm 1. Fix any constant $0 < \rho < 1$. Upon the arrival of a job j, job k can be replaced by job j if k is the smallest processing job, $p_k < p_j$ and job k has been processed no larger than ρ fraction.

Counter example. At $t = 0$, m identical jobs come with $p_1 = p_2 = \cdots = p_m = 1$. Each of them is scheduled on a machine. At $t = \rho$, job $(m+1)$ comes with $p_{m+1} = 1 + \xi$ (ξ is an infinitesimal amount). Then one of jobs 1 to m is replaced by job $(m+1)$. At $t = 2\rho$, job $(m+2)$ comes with $p_{m+2} = 1 + 2\xi$, then job $(m+1)$ is replaced by job $(m+2)$. The same thing goes on and on, and at time $t = 1$, job $(m - 1 + \lceil \frac{1}{\rho} \rceil)$ is replaced by job $(m + \lceil \frac{1}{\rho} \rceil)$. After then, at $t = 1$, $(m - \lceil \frac{1}{\rho} \rceil)$ jobs come with $p_{m+\lceil \frac{1}{\rho} \rceil + 1} = p_{m+\lceil \frac{1}{\rho} \rceil + 2} = \cdots = p_{2m} = 1$. Then there are m pending jobs but only $(m-1)$ idle machines, so $\mathsf{ALG} = 3$. In the optimal solution, no replacement happens, and $\mathsf{OPT} = 2 + \lceil \frac{1}{\rho} \rceil \xi$.

Candidate Algorithm 2. Fix $0 < \rho < 1$ and $1 < \mu < 2$. Upon the arrival of a job j, job k can be replaced by job j if $\mu p_k < p_j$ and job k has been processed no larger than ρ fraction.

We show a counter example using $\mu = 3/2$ and $\rho = 1/2$. This counter example can be generalized to any μ and ρ such that $\mu + \rho \leq 2$.

Counter example. At $t = 0$, m jobs come, with $p_1 = m+1, p_2 = m+2, \cdots, p_m = 2m$. At time $t = m$, job $(m+1)$ comes with $p_{m+1} = 3m$. Then, as job m is the only job that is processed at most half, job m is replaced by job $(m+1)$. After the replacement, $(m-1)$ jobs come with $p_{m+2} = 2m+1, p_{m+3} = 2m+2, \cdots, p_{2m} = 3m-1$. Then $\mathsf{ALG} = 6m$, while in the optimal solution, no replacement happens, thus $\mathsf{OPT} = 4m+1$.

Candidate Algorithm 3. Fix a target performance ratio $1+\gamma$. When a job j comes, schedule it virtually to ALG, and calculate the current optimal solution with all the jobs that have been released. If the ratio can still be bounded in $1+\gamma$, do not replace any jobs; otherwise choose one job to replace.

Counter example. At $t = 0$, m jobs comes first, with $p_1 = 2m, p_2 = 2m+1, \cdots, p_m = 3m-1$. After each of them has been scheduled on a machine, at $t = 0$, another m jobs come, with $p_{m+1} = 3m, p_{m+2} = 3m+1, \cdots, p_{2m} = 4m-1$. At this time, since the local ALG and local OPT are exactly the same, no replacement happens. Then at $t = 3m-1$, another job comes with $p_{2m+1} = 3m$. So for this instance, $\mathsf{ALG} = 6m-1$; while in the optimal solution, three smallest jobs (jobs 1, 2, 3) are scheduled on the same machine, while all other jobs are paired up with the smallest with largest, and $\mathsf{OPT} = 4m+3$.

For our algorithm LPT with Restart, some may wonder what happens if α or β is not in $(0, 1/2)$. we know that if $\alpha = 0$, it is exactly the same with LPT, which cannot beat 1.5; if $\beta = 0$, a counter example can be given that is similar to that of Candidate Algorithm 1. Next we show that when $\beta \geq 1/2$ or $\alpha \geq 1/2$, LPT with Restart could not beat 1.5, no matter what the value of the other parameter is.

Candidate Algorithm 4. In LPT with Restart, set $\beta \geq 1/2$, and α to be any constant.

Counter example. At $t = 0$, $m (m \geq 4)$ jobs come first, with $p_1 = p_2 = \cdots = p_m = 1$. After all these jobs are scheduled, still at time 0, another m jobs come, with $p_{m+1} = p_{m+2} = \cdots = p_{2m} = 3/2 + \xi$. Then at $t = 1$, another job comes with $p_{2m+1} = 2$. Since $\beta \geq 1/2$, no replacement happens, and $\mathsf{ALG} = 9/2 + \xi$, while in optimal schedule, all jobs could be completed at or before time $3 + \xi$.

Candidate Algorithm 5. In LPT with Restart, set $\alpha \geq 1/2$, and β to be any constant such that $0 < \beta < 1/2$.

Counter example. We consider the special case with only one machine. At $t = 0$, job 1 comes with $p_1 = 1$. At $t = 1 - \xi$ (again, ξ is an infinitesimal amount), job 2 comes with $p_2 = 2$, then job 1 is replaced by job 2. At $t = 3 - 2\xi$, job 3 comes with $p_3 = 4$, job 2 is replaced by job 3. The same thing goes on and on. Each time a new job comes, ALG would replace the previous job, while OPT would wait for the previous job to end. The final ratio would be arbitrarily close to 1.5.

D Hardness for deterministic algorithms with restart

In this section, we present a simple lower bound of $\sqrt{1.5} \approx 1.225$ for any deterministic algorithms with restart. Given any deterministic algorithm, consider the following instance with two machines.

At $t = 0$, two jobs come with size $p_1 = p_2 = 1$. Then, at $t = 3 - \sqrt{6}$, another job comes with size $p_3 = \sqrt{6} - 1$.

1. If the algorithm starts processing job 3 after time 1, i.e., after completing the two jobs of size 1, no more jobs arrive in the instance. We have $\mathsf{OPT} = 2$ as we could have scheduled job 1 and job 2 on the same machine and job 3 on the other one. On the other hand, $\mathsf{ALG} \geq 1 + p_3 = \sqrt{6}$.
2. If the algorithm starts processing job 3 before time 1, e.g., it restarts one of the size-1 jobs, let there be a fourth job that arrives at time 1 with size $p_4 = \sqrt{6} - 1$. We have $\mathsf{OPT} = \sqrt{6}$ by scheduling job 1 and 3 on one machine, and 2 and 4 on the other. On the other hand, we have $\mathsf{ALG} \geq 3$ since at time 1 at least one of jobs 1 and 2 is pending, and job 3 does not complete until time 2.

On Sketching the q to p Norms

Aditya Krishnan
Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, USA
arkrishn@andrew.cmu.edu

Sidhanth Mohanty
Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, USA
sidhanth@cmu.edu

David P. Woodruff
Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, USA
dwoodruf@cs.cmu.edu

Abstract

We initiate the study of data dimensionality reduction, or sketching, for the $q \to p$ norms. Given an $n \times d$ matrix A, the $q \to p$ norm, denoted $\|A\|_{q\to p} = \sup_{x\in\mathbb{R}^d\setminus\vec{0}} \frac{\|Ax\|_p}{\|x\|_q}$, is a natural generalization of several matrix and vector norms studied in the data stream and sketching models, with applications to datamining, hardness of approximation, and oblivious routing. We say a distribution S on random matrices $L \in \mathbb{R}^{nd} \to \mathbb{R}^k$ is a (k, α)-sketching family if from $L(A)$, one can approximate $\|A\|_{q\to p}$ up to a factor α with constant probability. We provide upper and lower bounds on the sketching dimension k for every $p, q \in [1, \infty]$, and in a number of cases our bounds are tight. While we mostly focus on constant α, we also consider large approximation factors α, as well as other variants of the problem such as when A has low rank.

2012 ACM Subject Classification Theory of computation → Numeric approximation algorithms

Keywords and phrases Dimensionality Reduction, Norms, Sketching, Streaming

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.15

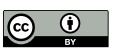
Related Version A full version of the paper is available at `https://arxiv.org/abs/1806.06429`.

Funding D. Woodruff would like to acknowledge the support by the National Science Foundation under Grant No. CCF-1815840.

1 Introduction

Data dimensionality reduction, or sketching, is a powerful technique by which one compresses a large dimensional object to a much smaller representation, while preserving important structural information. Motivated by applications in streaming and numerical linear algebra, the object is often a vector $x \in \mathbb{R}^n$ or a matrix $A \in \mathbb{R}^{n\times d}$. One of the most common forms of sketching is oblivious sketching, whereby one chooses a random matrix L from some distribution S, and compresses x to Lx or A to $L(A)$. The latter quantity $L(A)$ denotes a linear map from $\mathbb{R}^{nd}$, interpreting A as an nd-dimensional vector, to an often much lower dimensional space, say $\mathbb{R}^k$ for a value $k \ll nd$.

Sketching has numerous applications. For example, in the data stream model, one sees additive updates $x_i \leftarrow x_i + \Delta$, where the update indicates that x_i should change from its old value by an additive Δ. Given a sketch $L \cdot x$, one can update it by replacing it with $L \cdot x + \Delta \cdot L_{*,i}$, where $L_{*,i}$ denotes the i-th column of L. Thus, it is easy to maintain a sketch of a vector evolving in the streaming model. Similarly, in the matrix setting, given an



update $A_{i,j} \leftarrow A_{i,j} + \Delta$, one can update $L(A)$ to $L(A) + \Delta L(e_{i,j})$, where $e_{i,j}$ denotes the matrix with a single one in the (i,j)-th position, and is otherwise 0. If L is oblivious, that is, sampled from a distribution independent of x (or A in the matrix case), then one can create L without having to see the entire stream in advance. Other applications include distributed computing, whereby a vector or matrix is partitioned across multiple servers. For instance, server 1 might have a vector x^1 and server 2 a vector x^2. Given the sketches Lx^1 and Lx^2, by linearity one can combine them, using $L(x^1 + x^2) = Lx^1 + Lx^2$. In these applications it is important that the number k of rows of L is small, since it is proporational to the memory required of the data stream algorithm, or the communication in a distributed protocol. Here k is referred to as the *sketching* dimension.

Sketching vector norms is fairly well understood, and we have tight bounds up to logarithmic factors for estimating the ℓ_p-norms $\|x\|_p = (\sum_i |x_i|^p)^{1/p}$ for every $p \in [1, \infty]$; for a sample of such work, see [1, 10, 24, 23, 28, 27] for work in the related data stream context, and [40, 9, 33] for work specifically in the sketching model. Recently, there is work [13] characterizing the sketching complexity of any symmetric norm on a vector x. A number of works have also looked at sketching *matrix norms*. In particular, the Schatten p-norms $\|A\|_p = \left(\sum_{i=1}^{\text{rank}(A)} \sigma_i(A)^p\right)^{1/p}$ have gained considerable attention. They have proven to be considerably harder to approximate than the vector p-norms, and understanding their complexity has led to important algorithmic and lower bound techniques. A body of work has focused on understanding the complexity of estimating matrix norms in the data stream model with 1-pass over the stream [4, 34], as well as with multiple passes [15], the sketching model [32, 36], statistical models [31, 29], as well as the general RAM model [38, 44]. Dimensionality reduction in these norms also has applications in quantum computing [46, 22], and are studied in nearest neighbor search data structures [2].

1.1 Our Contributions

We consider the sketching complexity of a new family of norms, namely, the $p \to q$ norms of a matrix. A common quantity that arises in various applications is the amount by which a linear map A "stretches" vectors. One way to measure this quantity is the maximum singular value of A, which can be written as $\sup_{\|x\|_2=1} \|Ax\|_2$, and is just the Schatten-∞ norm, defined above. In this work we consider a different way of measuring this stretch, which considerably generalizes the operator norm.

For a linear operator A from a normed space $\mathcal{X}$ to a normed space $\mathcal{Y}$, we define $\|A\|_{\mathcal{X}\to\mathcal{Y}}$ as $\sup_{\|x\|_{\mathcal{X}}=1} \|Ax\|_{\mathcal{Y}}$. Of specific interest to us is the case where $\mathcal{X} = \ell_q^d$ and $\mathcal{Y} = \ell_p^n$, and we denote the corresponding norm of such an operator by $\|A\|_{q\to p}$. Our objective is to study the sketching complexity of approximating this norm.

▶ **Definition 1** ((k,α)-sketching family)**.** Let $\mathcal{S}$ be a distribution over linear functions from $\mathbb{R}^{n\times d}$ to $\mathbb{R}^k$ and f a function from $\mathbb{R}^k$ to $\mathbb{R}$. We call $(\mathcal{S}, f)$ a **(k,α)-sketching family** for the $q \to p$ norm if for all $A \in \mathbb{R}^{n\times d}$, $\mathbf{Pr}_{L\sim\mathcal{S}}\left[f(L(A)) \in (1/\alpha, \alpha)\,\|A\|_{q\to p}\right] \geq \frac{5}{6}$.

We provide upper and lower bounds on k. The details of the specific results we have are described in Section 1.3.

1.2 Motivation

This problem is well-studied in mathematics when $p = q$ as it simply corresponds to p-matrix norm estimation[1].

An intriguing question is whether one can preserve $\|Ax\|_p$ in a lower-dimensional sketch space, given that the vectors x come from the unit ball of a smaller norm.

Apart from being mathematically interesting, this problem has a number of applications. The operator norm is a special case when $p = q = 2$. The operator norm can be accurately estimated by any subspace embedding for ℓ_2, discussed in detail in [18]. The dual of this norm is also the Schatten-1 norm, which has received considerable attention in the streaming model [34, 15]. The $q \to p$ norm problem is a natural generalization of the operator norm problem, and when $p < 2$, may be more appropriate in the context of robust statistics, where it is known that the p norm for $p < 2$ is less sensitive to outliers, see, e.g., Chapter 3 of [47] for a survey on robust regression, and [42] for recent work on ℓ_1-low rank approximation.

The $2 \to q$ norms arise in the hardness of approximation literature and an algorithm for some instances of the problem was used to break the Khot-Vishnoi Unique Games candidate hard instance [30]. Work by [11] gives an algorithm running in time $\exp(n^{2/p})$ for approximating $2 \to p$ norms for all $p \geq 4$. These algorithms give a constant factor approximation when promised the $2 \to p$ norm is in a certain range (depending on the operator norm) rather than providing a general estimate of the $2 \to p$ norm. This same paper also discusses assumptions on the the NP-hardness and ETH hardness of approximating $2 \to p$ norms. The work of [14] extends that of [11] to all $p \geq 2$. The work of [12] gives a PTAS for computing $\|A\|_{q \to p}$ if $1 \leq p \leq q$ and A has non-negative entries, and gives an application of this to the oblivious routing problem where congestion is measured using the ℓ_p norm. The paper also shows that it is hard to approximate $\|A\|_{q \to p}$ within a constant factor for general A, and general p and q. Sketching may allow, for example, for reducing the original problem to a smaller instance of the same problem, which although may still involve exhaustive search, could give a faster concrete running time.

The $1 \to q$ norm turns out to be the maximum of the q-norm of the columns of A, which is related to the heavy hitters problems in data streams, e.g., the column with the largest q-norm may be the most significant or desirable in an application. Likewise, the $q \to \infty$ norms turn out to be the maximum of the p-norms of the rows of A, where p is the dual norm to q, and therefore have similar heavy hitter applications. The $\infty \to q$ norm is maximized when $x \in \{-1, 1\}^n$ and therefore includes the cut-norm as a special case, and is related to Grothendieck inequalities, see, e.g., [16, 39, 17].

Our main motivation for studying the $p \to q$ norms comes from understanding and developing new techniques for this family of norms. Another family of norms that is well-studied in the data stream literature are the *cascaded norms*, which for an $n \times d$ matrix A and parameters p and q, are defined to be $(\sum_{i=1,\ldots,n}(\|A_{i,*}\|_p)^q)^{1/q}$, where $A_{i,*}$ denotes the i-th row of A. That is, we compute the q-norm of the vector of p-norms of the rows of A. This problem originated in [19] and has applications to mining multi-graphs; the following sequence of work established tight bounds up to logarithmic factors for every $p, q \in [1, \infty]$ [26, 6]. This line of work led to very new techniques; one highlight is the use of Poincaré inequalities in proving information complexity lower bounds, which has then been studied in a number of followup works [5, 25, 7].

[1] See, e.g., `https://en.wikipedia.org/wiki/Matrix_norm`

1.3 Our Results

After establishing preliminary results and theorems in Section 2, we give our results for constant and large approximation factors. Our main theorem is as follows. Here ℓ_{q^*} is the dual norm of ℓ_q, that is, $1/q^* + 1/q = 1$ (when $q = 1$, $q^* = \infty$, and vice versa).

▶ **Theorem 2.** *For all matrices $A \in \mathbb{R}^{n\times n}$ with rank r and real values $p, q \in [1, \infty]$, the table below gives upper and lower bounds on k for a $(k, \Theta(1))$-sketching family of various $q \to p$ norms.*

$q \to p$ Norm	$p^* \to q^*$ Norm	Upper Bound	Sec	Lower Bound	Sec
$1 \to [1,2]$	$[2,\infty] \to \infty$	$O(n \log n)$	*3.1*	$\Omega(n)$	*4.2*
$1 \to [2,\infty]$	$[1,2] \to \infty$	$O(n^{2-\frac{2}{p}} \log^2 n)$	*3.1*	$\Omega(n^{2-\frac{2}{p}})$	*4.3*
$[2,\infty] \to [1,2]$	$[2,\infty] \to [1,2]$	$O(n^2)$	-	$\Omega(n^2)$	*4.4*
$2 \to [2,\infty]$	$[1,2] \to 2$	$O(\min\{n^{1-\frac{2}{p}} r^2 \log n, n^2\})$	*3.2*	$\Omega(\min\{n, n^{1-\frac{2}{p}} r\})$	*4.5*
$[1,2] \to [1,2]$	$[2,\infty] \to [2,\infty]$	$O(n^2)$	-	$\Omega(\min\{n^{1-\frac{2}{q^*}} r, n\})$	*4.5*
$[1,2] \to [2,\infty]$	$[1,2] \to [2,\infty]$	$O(n^2)$	-	$\Omega\left(\frac{n}{\log n}\right)$	*4.6*

The constant factor hidden in Theorem 2 does not hold for all constants, the smallest constant it holds for varies depending on the specific values of q, p.

We also have several results for large approximation factors summarized in the theorem below.

▶ **Theorem 3.** *There exists a $\left(O\left(\frac{n^2}{\alpha}\right), \alpha\right)$-sketching family for the $2 \to p$ and $\infty \to p$ norm and a $\left(O\left(\frac{n^2}{\alpha^2}\right), \alpha\right)$-sketching family for the $q \to p$ norm for $q \geq 1$ and $1 \leq p \leq 2$.*

Our algorithms combine several insights, which we illustrate here in the case of the $2 \to p$ norm for $p \geq 2$ and when the rank of A is r: (1) we show by duality that $\|A\|_{2\to p}$ is the same as $\|A^T\|_{p^*\to 2}$, where p^* satisfies $\frac{1}{p^*} + \frac{1}{p} = 1$ and is the dual norm to p. Although the proof is elementary, this plays several key roles in our argument. Next, we (2) use oblivious subspace embeddings S which provide constant factor approximations for all vectors simultaneously in an r-dimensional subspace of ℓ_2, and enable us to say that with Cr rows for a constant $C > 0$, we have $\|SA^T\|_{p^*\to 2} = \Theta(1)\|A^T\|_{p^*\to 2}$. Next, (3) we use that for a random Gaussian matrix $G \in \mathbb{R}^{C'r\times Cr}$, for a constant $C' > 0$, with appropriate variance, it has the property that simultaneously for all $x \in \mathbb{R}^{Cr}$, $\|Gx\|_1 = \Theta(1) \cdot \|x\|_2$. This is a special case of Dvoretsky's theorem in functional analysis. Thus, instead of directly approximating $\|SA^T\|_{p^*\to 2}$, we can obtain a constant factor approximation by approximating $\|GSA^T\|_{p^*\to 1}$. This is another norm we do not know how to directly work with, so we apply duality (1) again, and argue this is the same as approximating $\|AS^TG^T\|_{\infty\to p}$. A key observation is now (4), that $\sup_{x \text{ s.t. } \|x\|_\infty = 1} \|AS^TG^Tx\|_p$ is realized when x has each coordinate equal to 1 or -1. Consequently, as $x \in \mathbb{R}^{C'r}$, it suffices to use any sketch T for the p-norm of a fixed vector which fails with probability $\exp(-C'r)$, and estimate $\|TAS^TG^Tx\|_p$ for each of the $2^{C'r}$ possible maximizers x, and output the largest estimate. As there exist sketches T with $O(n^{1-2/p}r \log n)$ rows for this purpose, this gives us an overall sketching complexity of $O(n^{1-2/p}r^2 \log n)$.

We defer a discussion of our lower bound techniques to Section 4.

2 Preliminaries

In this section, we introduce the tools we use in this paper.

▶ **Definition 4** (Total Variation Distance)**.** Given two distributions $\mathcal{D}$ and $\mathcal{D}'$ over sample space Ω with density functions $p_{\mathcal{D}}$ and $p_{\mathcal{D}'}$, the **total variation distance** is defined in two equivalent ways as follows $d_{TV}(\mathcal{D}, \mathcal{D}') = \frac{1}{2}\|p_{\mathcal{D}} - p_{\mathcal{D}'}\|_1 = \sup_{\mathcal{E}} |\mathbf{Pr}_{x\sim\mathcal{D}}[\mathcal{E}] - \mathbf{Pr}_{x\sim\mathcal{D}'}[\mathcal{E}]|$

The following result bounds the total variation distance between two multivariate Gaussians.

▶ **Lemma 5** ([21], Lemma A4)**.** *Let λ be the minimum eigenvalue of PSD matrix Σ, then $d_{TV}(\mathcal{N}(\mu, \Sigma), \mathcal{N}(\mu', \Sigma')) \leq \frac{C}{\sqrt{\lambda}}(\|\mu - \mu'\|_2 + \|\Sigma - \Sigma'\|_F)$ for an absolute constant C.*

We state a well known result that a Lipschitz function of a Gaussian vector is tightly concentrated around its expectation, which is useful since ℓ_p norms are Lipschitz.

▶ **Theorem 6** ([43], Theorem 2.1.12)**.** *Let $X \sim \mathcal{N}(0, I_n)$ be a Gaussian random vector and let $f : \mathbb{R}^n \to \mathbb{R}$ be a 1-Lipschitz function. Then for some absolute constants $C, c > 0$, $\mathbf{Pr}[|f(X) - \mathbf{E}[f(X)]| \geq \lambda] \leq C\exp(-c\lambda^2)$ Notice that this implies if f is t-Lipschitz, then $\mathbf{Pr}[|f(X) - \mathbf{E}[f(X)]| \geq \lambda] \leq C\exp(-c\lambda^2/t^2)$*

It is possible to embed ℓ_2^n into $\ell_p^{O(n)}$ with constant distortion using a linear map when $p \in [1, 2]$, and we use the existence of such a linear map in our results.

▶ **Lemma 7** ([37], Theorem 2.5.1)**.** *For all $p \in [1, 2]$, there is an absolute constant C_p such that for any n, there is a linear map $T : \mathbb{R}^n \to \mathbb{R}^{C_p n}$ such that $\|T(x)\|_p = \left(1 \pm \frac{1}{2}\right)\|x\|_2$. An important observation is that this implies for any linear map $A : \mathbb{R}^n \to \mathbb{R}^n$, we have $\|TA\|_{q\to p} = \left(1 \pm \frac{1}{2}\right)\|A\|_{q\to 2}$.*

In the lemma below we make an important observation that highlights the connection between several $p \to q$ norms.

▶ **Lemma 8.** *For any $p, q \geq 1$ and $d \times n$ matrix A, $\|A\|_{q\to p} = \|A^T\|_{p^*\to q^*}$.*

Proof. Using the notation above for dual norms, we have

$$\begin{aligned}
\|A\|_{q\to p} &= \sup\{\|Ax\|_q : \|x\|_p \leq 1\} \\
&= \sup\{\sup\{y^\top Ax : \|y\|_{q*} \leq 1\} : \|x\|_p \leq 1\} \\
&= \sup\{\sup\{x^\top A^\top y : \|x\|_p \leq 1\} : \|y\|_{q^*} \leq 1\} \\
&= \sup\{\|A^\top y\|_{p^*} : \|y\|_{q^*} \leq 1\} \\
&= \|A^\top\|_{p^*\to q^*}
\end{aligned}$$

◀

Throughout the paper, we make use of q^* to refer to $\frac{q}{q-1}$ since $\ell_{\frac{q}{q-1}}$ is the dual norm of ℓ_q.

We give a characterization of the $1 \to p$ and $\infty \to p$ norm of a matrix. The proofs can be found in the full version's Appendix A. For any $d \times n$ matrix A, we have

▶ **Lemma 9.** $\|A\|_{1\to p} = \max_{i\in[n]}\{\|A_{*,i}\|_p\}$.

▶ **Lemma 10.** $\|A\|_{\infty\to p} = \max_{x\in\{\pm 1\}^n}\|Ax\|_p$.

We introduce the machinery of ε-nets, a common tool in the study of random matrices (see [45]) along with some relevant lemmas and defer the proofs to the full version's Appendix.

▶ **Definition 11** (ε-net)**.** Let $\mathcal{X}$ be a normed space. For $S \subseteq V$, we call a set N an ε-net for S if for all $v \in S$, there is $v' \in N$ such that $\|v - v'\|_{\mathcal{X}} < \varepsilon$.

For a linear operator A, we show that to bound $\|A\|_{\mathcal{X}\to\mathcal{Y}}$, it suffices to bound $\|Ax\|_{\mathcal{Y}}$ for x taken over an ε-net of the unit ball in $\mathcal{X}$.

▶ **Lemma 12.** *Let $\mathcal{X}$ and $\mathcal{Y}$ be normed spaces and let $A : \mathcal{X} \to \mathcal{Y}$ be a linear map. Suppose N is an ε-net of the unit ball in $\mathcal{X}$, then $\|A\|_{\mathcal{X}\to\mathcal{Y}} \leq \frac{1}{1-\varepsilon} \max_{v\in N} \|Av\|_{\mathcal{Y}}$.*

We also give a way to construct 'small' ε-nets of unit balls.

▶ **Lemma 13.** *There is an ε-net of the unit ball B in an n-dimensional normed space $\mathcal{X}$ with at most $\left(\frac{2+\varepsilon}{\varepsilon}\right)^n$ elements.*

Another tool we use is subspace embeddings, which we define below.

▶ **Definition 14.** An **oblivious subspace embedding family** (OSE family) is a distribution $\mathcal{S}$ over $O(m) \times n$ matrices such that for any subspace $K \subseteq \mathbb{R}^n$ of dimension m, $\mathbf{Pr}_{S\sim\mathcal{S}}[\forall x \in K : \|Sx\|_2 = \Theta(1)\|x\|_2] \geq \frac{9}{10}$.

▶ **Lemma 15** ([41])**.** *There exist OSE families, where the matrices have dimension $O(k) \times n$. Note that this means for any rank-k matrix A, a randomly drawn S from such an oblivious subspace embedding family satisfies $\|SAx\|_2 = \Theta(1)\|Ax\|_2$ simultaneously for all x with probability at least $99/100$.*

3 Sketching algorithms for constant factor approximations

3.1 Sketches for approximating $\|A\|_{1\to p}$

We show how to use sketches for p-norms of vectors to come up with sketches for the $1 \to p$ norm.

▶ **Lemma 16.** *Let x be an arbitrary vector in $\mathbb{R}^n$. If $\mathcal{S}$ is a distribution over $t \times n$ sketching matrices, and $f : \mathbb{R}^t \to \mathbb{R}$ is a function such that $\mathbf{Pr}_{S\sim\mathcal{S}}\left[f(Sx) \in \left(\frac{1}{2}\|x\|_p, 2\|x\|_p\right)\right] \geq \frac{2}{3}$ then there is an $(O(nt\log n), 2)$-sketching family $(\mathcal{S}', g)$ for the $1 \to p$ norm of $n \times n$ matrices.*

Proof. Proof in the full version's Appendix B. ◀

Given an n-dimensional vector x, we have the following theorems from [28] and [6] respectively.

▶ **Theorem 17** (Efficient sketches for small norms)**.** *When $p \in [1, 2]$, there is a function f and a distribution over sketching matrices $\mathcal{F}$ with $O(1)$ rows such that for $S \sim \mathcal{F}$, $f(Sx)$ is a constant factor approximation for $\|x\|_p$ with probability at least $2/3$.*

▶ **Theorem 18** (Efficient sketches for large norms)**.** *When $p > 2$, there is a function f and a distribution over sketching matrices $\mathcal{F}$ with $O(n^{1-2/p}\log n)$ rows such that for $S \sim \mathcal{F}$, $f(Sx)$ is a constant factor approximation for $\|x\|_p$ with probability at least $2/3$.*

Lemma 16 tells us the following as a corollary to Theorems 17 and 18.

▶ **Theorem 19.** *There is an $(O(n\log n), 2)$-sketching family for the $1 \to p$ norm when $p \in [1, 2]$ and a $(O(n^{2-2/p})\log^2 n, 2)$-sketching family for the $1 \to p$ norm when $p \in (2, \infty]$.*

3.2 Sketches for approximating $\|A\|_{2\to p}$ for $p > 2$

We give a sketching algorithm for the $2 \to p$ norm of A, whose number of measurements depends on the rank r of $d \times n$ matrix A.

▶ **Theorem 20.** *There is an $(O(n^{1-2/p}r^2\log n), \Theta(1))$-sketching family for the $2 \to p$ norm.*

Proof. Observe that $\|A\|_{2\to p}$ is equal to $\|A^T\|_{p^*\to 2}$ by Lemma 8 and let S be a $Cr \times d$ matrix drawn from an oblivious subspace embedding family, which exists by Lemma 15. From Lemma 7, let G be a $\beta r \times Cr$ map such that for all x, $\|GSA^Tx\|_1 = \Theta(1)\|SA^Tx\|_2$. Combining with the subspace embedding property, we get that $\|GSA^Tx\|_1 = \Theta(1)\|A^Tx\|_2$ for all x, which is equivalent to saying $\|GSA^T\|_{p^*\to 1} = \Theta(1)\|A\|_{2\to p}$. Another application of Lemma 8 gives us that $\|AS^TG^T\|_{\infty\to p} = \Theta(1)\|A\|_{2\to p}$. Since AS^TG^T is $n \times \beta r$, $\|AS^TG^T\|_{\infty\to p} = \max_{x\in\{\pm 1\}^{\beta r}} \|AS^TG^Tx\|_p$.

Our final ingredient is the existence of an $O(n^{1-2/p}\log n \log(1/\delta)) \times n$ sketching matrix E and estimation function f such that for any x, $\mathbf{Pr}[f(Ey) = \Theta(1)\|y\|_p] \geq 1-\delta$ [3] when $p > 2$. We set $\delta = 2^{-2\beta r}$ and use a union bound over all $2^{\beta r}$ vectors in $\{\pm 1\}^{\beta r}$ to conclude

$$\mathbf{Pr}[\forall x \in \{\pm 1\}^{\beta r} : f(EAS^TG^Tx) = \Theta(1)\|AS^TG^Tx\|_q] \geq 1 - 2^{-\beta r}$$

$$\mathbf{Pr}\left[\max_{x\in\{\pm 1\}^{\beta r}} f(EAS^TG^Tx) = \Theta(1)\|AS^TG^T\|_{\infty\to q}\right] \geq 1 - 2^{-\beta r}$$

Consequently, we get a sketch that consists of $O(n^{1-2/p}r^2\log n)$ measurements to get a $\Theta(1)$ approximation to $\|A\|_{2\to p}$ with probability at least 0.99. ◀

4 Sketching lower bounds for constant factor approximations

4.1 Lower Bound Techniques

The way we prove most of our lower bounds is by giving two distributions over $n \times n$ matrices, $\mathcal{D}_1$ and $\mathcal{D}_2$, where matrices drawn from the two distributions have $q \to p$ norm separated by a constant factor κ with high probability, which means a $(k, \sqrt{\kappa})$-sketching family can distinguish between samples from the two distributions. We then show an upper bound on the variation distance between distributions of k-dimensional sketches of $\mathcal{D}_1$ and $\mathcal{D}_2$. We then argue that if k is too small, then the total variation distance is too small to solve the distinguishing problem. We formalize this intuition in the following theorem.

▶ **Theorem 21.** *Suppose $\mathcal{D}_1$ and $\mathcal{D}_2$ are distributions over $d \times n$ matrices such that*

(i) $\mathbf{Pr}_{D\sim\mathcal{D}_1}[\|D\|_{q\to p} < s] \geq 1 - \frac{1}{n}$ *and* $\mathbf{Pr}_{D\sim\mathcal{D}_2}[\|D\|_{q\to p} > \kappa s] \geq 1 - \frac{1}{n}$

(ii) *for any linear map $L : \mathbb{R}^{d\times n} \to \mathbb{R}^k$, $d_{TV}(L(\mathcal{D}_1), L(\mathcal{D}_2)) = O\left(\frac{k^a}{n^b}\right)$*

for constants s, κ, a, b, any $(k, \sqrt{\kappa})$-sketching family for the $q \to p$ norm must satisfy $k = \Omega(n^{b/a})$.

Proof. Let $\mathcal{D}$ be the distribution over matrices given by sampling from $\mathcal{D}_1$ with probability $\frac{1}{2}$ and drawing from $\mathcal{D}_2$ with probability $\frac{1}{2}$. We shall fix a sketching operator $L : \mathbb{R}^{d\times n} \to \mathbb{R}^k$ and consider A drawn from a distribution $\mathcal{D}$. Suppose $f(L(A))$ lies in $(1/\sqrt{\kappa}, \sqrt{\kappa})\|A\|_{q\to p}$ with probability at least $5/6$. It suffices to show that k must be $\Omega(n^{b/a})$ since the theorem statement then follows from Yao's minimax principle. We must have

$$\mathbf{Pr}_{A\sim\mathcal{D}_1}\left[f(L(A)) \in \left(\frac{1}{\sqrt{\kappa}}, \sqrt{\kappa}\right)\|A\|_{q\to p}\right] \geq \frac{2}{3},$$

$$\mathbf{Pr}_{A\sim\mathcal{D}_2}\left[f(L(A)) \in \left(\frac{1}{\sqrt{\kappa}}, \sqrt{\kappa}\right)\|A\|_{q\to p}\right] \geq \frac{2}{3}$$

Thus, we have an algorithm that correctly distinguishes with probability at least $\frac{3}{5}$ if A was drawn from $\mathcal{D}_1$ or $\mathcal{D}_2$ by checking if $f(L(A))$ is greater than or less than $\sqrt{\kappa}s$.

The existence of this distinguishing algorithm means the total variation distance between the distributions of $L(D_1)$ and $L(D_2)$ is at least $\frac{1}{5}$. From the theorem's hypothesis, we know of a constant C such that $\frac{Ck^a}{n^b} \geq \frac{1}{5}$, which gives us the desired upper bound. ◀

We also show an upper bound on the variation distance of sketches for two distributions that we use throughout this paper. Define $\mathcal{G}_{1,d\times n}$ as the distribution over $d \times n$ Gaussian matrices and $\mathcal{G}_{2,d\times n}[\alpha]$ as the distribution given by drawing a Gaussian matrix and adding αu, where u is a d-dimensional Gaussian vector to a random column. We write $\mathcal{G}_i$ instead of $\mathcal{G}_{i,d\times n}$ when the dimensions of the random matrix are evident from context.

▶ **Lemma 22.** *Let L be a linear sketch from $\mathbb{R}^{d\times n} \to \mathbb{R}^k$ and let $\mathcal{H}_i$ be the distribution of $L(x)$ where x is drawn from $\mathcal{G}_i$. Then $d_{TV}(\mathcal{H}_1, \mathcal{H}_2) \leq \frac{C\alpha^2 k}{n}$ for an absolute constant C.*

Proof. We can think of L as a $k \times nd$ matrix that acts on a sample from $\mathcal{G}_1$ or $\mathcal{G}_2$ as though it were an nd-dimensional vector. Without loss of generality, we can assume that the rows of L are orthonormal, since one can always perform a change of basis in post-processing. Thus, the distribution $\mathcal{H}_1$ is the same as $\mathcal{N}(0, I_k)$. For fixed i and G a $d \times n$ matrix of unit Gaussians, the distribution of $L(G + \alpha u e_i^T)$ is Gaussian with covariance $\mathbf{E}[L(G + \alpha u e_i^T)L(G + \alpha u e_i^T)^T]$, equal to $I + \alpha^2 L_{B_i} L_{B_i}^T$ where L_{B_i} is the submatrix given by columns of L indexed $(i-1)d+1, (i-1)d+2, \ldots, id$. Let $\mathcal{H}_{2,i}$ be $\mathcal{N}(0, I + \alpha^2 L_{B_i} L_{B_i}^T)$. $\mathcal{H}_2$ is the distribution of picking a random i and drawing a matrix from $\mathcal{N}(0, I + L_{B_i} L_{B_i}^T)$.

We now analyze the total variation distance between $\mathcal{H}_1$ and $\mathcal{H}_2$ and get the desired bound from a chain of inequalities. $d_{TV}(\mathcal{H}_1, \mathcal{H}_2) = \frac{1}{2}\int_{x\in\mathbb{R}^k} |p_{\mathcal{H}_1}(x) - p_{\mathcal{H}_2}(x)|dx$

$\leq \frac{1}{2}\int_{x\in\mathbb{R}^k} \left|\sum_{i=1}^n \frac{1}{n} p_{\mathcal{H}_1}(x) - \frac{1}{n} p_{\mathcal{H}_{2,i}}(x)\right| dx \leq \frac{1}{n}\sum_{i=1}^n \frac{1}{2}\int_{x\in\mathbb{R}^k} \left|p_{\mathcal{H}_1}(x) - p_{\mathcal{H}_{2,i}}(x)\right| dx$

$\leq \frac{1}{n}\sum_{i=1}^n d_{TV}(\mathcal{N}(0, I_k), \mathcal{H}_{2,i}) \leq \frac{1}{n}\sum_{i=1}^n C\alpha^2 \|L_{B_i} L_{B_i}^T\|_F \leq \frac{1}{n}\sum_{i=1}^n C\alpha^2 \|L_{B_i}\|_F^2$

$\leq \frac{C\alpha^2}{n}\|L\|_F^2 = \frac{C\alpha^2 k}{n}$. The third last inequality follows from Lemma 5. ◀

4.2 Lower bounds for approximating $\|A\|_{1\to p}$ for $1 \leq p \leq 2$

We follow the lower bound template given in Section 4.1.

▶ **Lemma 23.** *For any κ, there exist values s_p such that with probability at least $1 - 1/n$, $\|G_1\|_{1\to p} \leq s_p$ and $\|G_2\|_{1\to p} \geq \kappa s_p$, for $1 \leq p \leq 2$, and $G_1 \sim \mathcal{G}_1$ and $G_2 \sim \mathcal{G}_2[\kappa]$.*

Proof. Recall that from Section 3.1, we know that $\|A\|_{1\to p} = \max_{i\in[n]} \|A_{*,i}\|_p$ which means that it suffices to give bounds on the maximum ℓ_p norm across columns of G_1 and G_2 respectively.

The ℓ_p norm is ζ_p-Lipschitz, where ζ_p is equal to $n^{1/p-1/2}$ in the regime $1 \leq p \leq 2$. For a given vector of standard Gaussians g, the probability that $\|g\|_p$ deviates from $\mathbf{E}[\|g\|_p]$ by more than $\beta\zeta_p\sqrt{\log n}$ is at most $C'e^{-c\beta^2 \log n}$ from Theorem 6 where C' is the constant C from the theorem, which for large enough choice of β can be made smaller than $1/n^2$. By a union bound over all columns, the probability that $\|G_1\|_{1\to p}$ exceeds $\mathbf{E}[\|g\|_p] + \beta\zeta_p\sqrt{\log n}$ is at most $1/n$. On the other hand, consider the perturbed column vector of G_2, which we denote g'. The probability that $\|g'\|_2$ is smaller than $\mathbf{E}[\|g'\|_p] - \beta\sqrt{1+\kappa^2}\zeta_p\sqrt{\log n} = \sqrt{1+\kappa^2}(\mathbf{E}[\|g\|_p] - \beta\zeta_p\sqrt{\log n})$ is at most $1/n^2$ by appropriate choice of β and Theorem 6, from which a lower bound on $\|G_2\|_{1\to p}$ that holds with probability at least $1 - \frac{1}{n^2}$ immediately follows.

Since $\mathbf{E}[\|g\|_p]$ is $\Theta(n^{1/p})$ and the deviations from expectations in upper bounds on $\|G_1\|_{1\to p}$ and lower bounds on $\|G_2\|_{1\to p}$ are asymptotically less than the expectations. ◀

The desired theorem is immediate from Lemma 23, Lemma 22, and Theorem 21 using $\mathcal{D}_1 = \mathcal{G}_{1,n\times n}$, and $\mathcal{D}_2 = \mathcal{G}_2[\kappa]$.

▶ **Theorem 24.** *Suppose $p \in [1, 2]$ and $(\mathcal{S}, f)$ is a $(k, \sqrt{\kappa})$-sketching family for the $1 \to p$ norm where κ is some constant, then $k = \Omega(n)$.*

4.3 Lower bound for approximating $\|A\|_{1\to p}$ for $p > 2$

We follow the lower bound template given in Section 4.1.

Denote $\mathbf{E}[\|g\|_p]$ as η_p. Let $\mathcal{G}_1$ be the distribution over $n \times n$ matrices given by i.i.d. Gaussians, and $\mathcal{G}_2[\alpha, \eta_p]$ be the distribution over $n \times n$ matrices given by taking a Gaussian matrix and adding $\alpha\eta_p$ to a random entry.

Since the proofs are very similar to those in Sections 4.1 and 4.2. We defer them to the full version's Appendix C.1.

▶ **Lemma 25.** *For any κ, there exists s_p such that with probability at least $1 - \frac{1}{n}$, $\|G_1\|_{1\to p} \leq s_p$ and $\|G_2\|_{1\to p} \geq \kappa s_p$, such that $G_1 \sim \mathcal{G}_1$ and $G_2 \sim \mathcal{G}_2[C\kappa, \eta_p]$ for some absolute constant C and $p > 2$.*

▶ **Lemma 26.** *Let L be a linear sketch from $\mathbb{R}^{n\times n} \to \mathbb{R}^k$ and let $\mathcal{D}_i$ be the distribution of $L(x)$ where x is drawn from $\mathcal{G}_i$. Then $d_{TV}(\mathcal{D}_1, \mathcal{D}_2) \leq \frac{C'\alpha\eta_p\sqrt{k}}{n}$ for an absolute constant C'.*

The theorem below immediately follows from Lemma 25, Lemma 26 and Theorem 21 using $\mathcal{D}_1 = \mathcal{G}_1$ and $\mathcal{D}_2 = \mathcal{G}_2[C\kappa, \eta_p]$.

▶ **Theorem 27.** *Suppose $(\mathcal{S}, f)$ is a (k, κ)-approximate sketching family for the $1 \to p$ norm for $p > 2$ and some constant κ, then $k = \Omega\left(\frac{n^2}{\eta_p^2}\right)$. In particular, using the fact that η_p is $\Theta(n^{1/p})$ for $p < \infty$ and $\Theta(\sqrt{\log n})$ when $p = \infty$ gives $k = \Omega\left(n^{2-\frac{2}{p}}\right)$ when $p < \infty$ and $k = \Omega\left(\frac{n^2}{\log n}\right)$ when $p = \infty$.*

4.4 Lower bound for approximating $\|A\|_{q\to p}$ when $q \geq 2$ and $p \leq 2$

We use the known lower bound of $\Omega(n^2)$ for sketching the $2 \to 2$ norm from [35] to deduce a lower bound on sketching the $q \to p$ norm for $q \geq 2$ and $p \leq 2$.

▶ **Theorem 28.** *Suppose $q \geq 2$ and $p \leq 2$, and if $(\mathcal{S}, f)$ is a $(k(n), \gamma)$-approximate sketching family for the $q \to p$ norm where γ is some constant, then $k(n) = \Omega(n^2)$.*

Proof. We prove this by showing that if the hypothesis of the theorem statement holds, then the $2 \to 2$ norm can be sketched in $O(k)$ measurements.

Given an $n \times n$ matrix A for which we want to sketch the $2 \to 2$ norm, note that by Lemma 7 there is a $Cn \times n$ matrix L_1 such that $\|L_1A\|_{2\to q^*} = (\frac{1}{\beta}, \beta)\|A\|_{2\to 2}$ for a constant β, and by Lemma 8 $\|L_1A\|_{2\to q^*} = \|A^TL_1^T\|_{q\to 2}$, and another application of Lemma 7 gives us another $Cn \times n$ matrix L_2 for which $\|L_2A^TL_1^T\|_{q\to p} = (\frac{1}{\beta}, \beta)\|A^TL_1^T\|_{q\to 2}$. Note that this means $\|L_2A^TL_1^T\|_{q\to p} = \left(\frac{1}{\beta^2}, \beta^2\right)\|A\|_{2\to 2}$, so we can sketch A by drawing a random L from $\mathcal{D}$ and storing $L(L_2A^TL_1^T)$, which uses $k(Cn)$ measurements and serves as a sketch from which f can be used to estimate $\|A\|_{2\to 2}$ within a constant factor, which means from [35], $k(Cn)$ must be $\Omega(n^2)$, which means $k(n) = \Omega(n^2/C^2) = \Omega(n^2)$. ◀

4.5 Lower bounds for approximating $\|A\|_{q\to p}$ for $p, q \leq 2$ and $p, q \geq 2$

In this section, we show a lower bound on the sketching complexity of $\|A\|_{q\to p}$ where A is a rank r matrix, when both p and q are at most 2. A corresponding lower bound for when p and q are at least 2 follows from Lemma 8. We achieve this by first showing a lower bound on the sketching complexity of $\|A\|_{2\to q}$ and then use Dvoretzky's theorem along with the relation between the $q \to p$ norm and the $p^* \to q^*$ norm to deduce the result.

We show a lower bound for sketching the $2 \to q$ norm using the template from Section 4.1. We use distributions $\mathcal{D}_1 = \mathcal{G}_{1,r\times n}$ and $\mathcal{D}_2[\alpha] = \mathcal{G}_{2,r\times n}\left[\alpha\frac{d}{\sqrt{r}}\right]$, as defined in Section 4.1 where d is $\max\{n^{1/q}, \sqrt{r}\}$.

▶ **Lemma 29.** *There exist values s_q and t_q such that with high probability, $\|G_1\|_{2\to q} \le s_q$ and $\|G_2\|_{2\to q} \ge C\alpha s_q$ for some absolute constant C, for $q > 2$, and $G_1 \sim \mathcal{D}_1$ and $G_2 \sim \mathcal{D}_2[\alpha]$.*

Proof. Let N be a 1/3-net of the Euclidean ball in $\mathbb{R}^r$ with 7^r elements, which exists by Lemma 13. For a fixed $x \in N$, G_1x is distributed as an n-dimensional vector with independent Gaussians, whose q-norm is at most $\beta_1 n^{1/q}$ for some constant β_1 in expectation and exceeds $\beta_1 n^{1/q} + \beta_2\sqrt{r}$ with probability at most $\frac{1}{8^r}$ for appropriate constant β_2, which follows from the q-norm being 1-Lipschitz and Theorem 6. A union bound over all $x \in N$ implies that with probability at least $1 - (7/8)^r$, $\forall x \in N : \|G_1x\|_q \le \beta_1 n^{1/q} + \beta_2\sqrt{r}$.

Then by applying Lemma 12, we conclude that with probability at least $1 - (7/8)^r$, $\|G_1\|_{2\to q} \le \frac{3}{2}(\beta_1 n^{1/q} + \beta_2\sqrt{r}) \le \frac{3}{2}(\beta_1 + \beta_2)d$. On the other hand, the perturbed row of G_2, called g' is distributed as $\sqrt{1 + \alpha^2\frac{d^2}{r}}g$ for a vector of i.i.d. Gaussians g. If we take the unit vector u in the direction of g', then the entry of G_2u corresponding to the perturbed row is concentrated around $\sqrt{1 + \alpha^2\frac{d^2}{r}}\|g\|_2 = \sqrt{r + \alpha^2 d^2}$, which means $\|G_2\|_{2\to q} \ge (1 - o(1))\sqrt{r + \alpha^2 d^2} \ge 0.9\alpha d$ with high probability. ◀

The theorem below immediately follows from Lemma 29, Lemma 22 and Theorem 21.

▶ **Theorem 30.** *Suppose $q \ge 2$ and $(\mathcal{S}, f)$ is a (k, γ)-sketching family for the $2 \to q$ norm of rank r matrices for some constant γ. Then $k = \Omega(nr/d^2)$.*

▶ **Theorem 31.** *Suppose $p, q \le 2$ and $(\mathcal{S}, f)$ is a (k, γ)-sketching family for the $q \to p$ norm of rank r matrices for some constant γ. Then $k = \Omega(nr/d^2)$ where $d = \max\{\sqrt{r}, n^{1/q^*}\}$.*

Proof. For a matrix A, from Lemma 8 we have that $\|A\|_{2\to q^*} = \|A^T\|_{q\to 2}$, and from Lemma 7, we know there is a $Cr \times r$ matrix L_1 such that $\|L_1A^T\|_{q^*\to p} = \Theta(1)\|A\|_{2\to q^*}$. We can use $(\mathcal{S}, f)$ to sketch L_1A^T to obtain an $(O(k), \Theta(1))$-sketching family for the $2 \to q^*$ norm, whose lower bound from Theorem 30 gives us the desired lower bound. ◀

4.6 Lower bounds for approximating $\|A\|_{q\to p}$ for $1 \le q \le 2$ and $p \ge 2$

We prove the desired lower bound using the template from Section 4.1. Let $\mathcal{D}_1$ be a distribution over $n \times n$ matrices where diagonal entries are Gaussians and off-diagonal entries are 0 and let $\mathcal{D}_2[\alpha]$ be a distribution over $n \times n$ matrices where a matrix is drawn from $\mathcal{D}_1$ and $\alpha\sqrt{\log n}$ is added to a random diagonal entry.

▶ **Lemma 32.** *There exists values $s_{p,q}$, $t_{p,q}$ and α such that with probability at least $1 - 1/n$, $\|G_1\|_{q\to p} \le s_{p,q}$ and $\|G_2\|_{q\to p} \ge \kappa s_{p,q}$ for some desired constant factor κ separation, such that $G_1 \sim \mathcal{D}_1$ and $G_2 \sim \mathcal{D}_2[\alpha]$.*

We give the proof of Lemma 32 in the full version's Appendix C.2.

Without loss of generality, we can assume that any sketch of G_1 and G_2 acts on $\mathrm{diag}(G_1)$ and $\mathrm{diag}(G_2)$ respectively. Lemma 26 gives an upper bound of $O(\sqrt{k\log n}/\sqrt{n})$ on the variation distance between k-dimensional sketches of these distributions. Thus, from the variation distance bound, Lemma 32 and Theorem 21, the desired theorem follows.

▶ **Theorem 33.** *Suppose $q \ge 2$ and $(\mathcal{S}, f)$ is a (k, γ)-sketching family for the $q \to p$ norm of rank r matrices for some constant γ, then $k = \Omega(n/\log n)$.*

5 Sketching with large approximation factors

While our results primarily involve constant factor approximations, we give several preliminary results studying large approximation factors for sketching the important cases of the $2 \to q$ norm and $[1, \infty] \to [1, 2]$ norms. Our goal is, given an approximation factor $\alpha(n)$, to give upper and lower bounds on k for a $(k, \alpha(n))$-sketching family for the respective norms. As a shorthand, we will refer to $\alpha(n)$ as α.

5.1 Sketching upper bounds for large approximations of $\|A\|_{2\to q}$

It is sufficient to give a (k, α)-sketching family for the $\infty \to q$ norm. To see why, given an input matrix $A \in \mathbb{R}^{n\times n}$, by Lemma 8 we have that $\|A\|_{2\to q} = \|A^T\|_{q^*\to 2}$. Using Lemma 7, there is a linear map such that this is equal within a constant factor of $\|GA^T\|_{q^*\to 1} = \|AG^T\|_{\infty\to q}$.

▶ **Theorem 34.** *Given a matrix $A \in \mathbb{R}^{n\times n}$, there exists a $(O(\frac{n^2}{\alpha}), \alpha)$-sketching family given by $(\mathcal{S}, f)$ for the $\infty \to q$ norm.*

Proof. Let $B \in \mathbb{Z}^+$ be some positive integer to be chosen later. Let the columns of our sketch matrix S be indexed by sets given by $\{B_i\}_{i=1}^{n/B}$ such that $B_i = ((i-1)B, iB]$. For each column v_{B_i}, we define i.i.d random variables $\{\sigma_{ij}\}_{j=1}^{B}$ such that $\sigma_{ij} = 1$ with probability $\frac{1}{2}$ and -1 with probability $\frac{1}{2}$. Let the column v_{B_i} be as follows:

$$v_{B_i}[j] = \begin{cases} \sigma_{ij} & \text{for } j \in [(i-1)B, iB] \\ 0 & \text{o/w} \end{cases}$$

We define our linear map $L(A)$ to be $L(A) = AS$. Our function $f : \mathbb{R}^{n/B} \to \mathbb{R}$ simply optimizes over $\{-1, 1\}^{n/B}$ and outputs $\|AS\|_{\infty\to q}$.

Since all $\sigma_{ij} \in \{-1, 1\}$ we have that $f(L(A)) \le \|A\|_{\infty\to q}$ since Sx for $x \in \{-1, 1\}^{n/B}$ has the property that $Sx \in \{-1, 1\}^n$.

We now show a lower bound on $f(L(A))$. To do so, we let T_i denote the column indices of A such that the index is column i in its respective block. We then notice that there exists $i \in [n/B]$ such that $\|A_{*,T_i}\|_{\infty\to q} \ge \frac{B}{n}\|A\|_{\infty\to q}$. We get this by applying the triangle inequality $\|A\|_{\infty\to q} \le \sum_{i=1}^{n/B} \|A_{*,T_i}\|_{\infty\to q}$.

Let i^* be the index that realizes this n/B-approximation to $\|A\|_{\infty\to q}$ and let $\{s_1\}_{i=1}^{n/B}$ be the assignment of signs that realizes the $\infty \to q$ norm of $A_{*,T_{i^*}}$.

$$f(L(A)) \ge \|\sum_{i=1}^{B}\sum_{j=1}^{n/B} s_j A_{*,B_j[i]}\|_q \ge \|\underbrace{\sum_{j=1}^{n/B} s_j A_{*,B_j[i^*]}}_{y} + \underbrace{\sum_{i\neq i^*}^{B}\sum_{j=1}^{n/B} s_j A_{*,B_j[i]}}_{z}\|_q$$

Notice that z is symmetric around the origin and hence we get that $\|y + z + y - z\|_q \le \frac{\|y+z\|_q + \|y-z\|_q}{2}$ which implies that $f(L(A)) \ge \|y + z\|_q \ge \Theta(1)\|y\|_q \ge \frac{n}{B}\|A\|_{\infty\to q}$ with probability at least $\frac{1}{2}$. Thus, we get an $O\left(\frac{n^2}{\alpha}\right)$ space sketch that gives us an α-approximation by setting $B = n/\alpha$. ◀

5.2 Sketching upper bounds for large approximations of $\|A\|_{q\to p}$ for $q \in [1, \infty]$ and $p \in [1, 2]$

We give a description of our sketch followed by the approximation factor. Towards the end of defining our sketch, let $B \in \mathbb{Z}^+$ be some positive integer to be chosen later. Let the rows of our sketch matrix S be indexed by sets given by $\{B_i\}_{i=1}^{n/B}$ such that $B_i = ((i-1)B, iB]$. For

each row v_{B_i}, we define i.i.d random variables $\{\sigma_{ij}\}_{j=1}^{B}$ such that $\sigma_{ij} = 1$ with probability $\frac{1}{2}$ and -1 with probability $\frac{1}{2}$. Let the row v_{B_i} be as follows:

$$v_{B_i}[j] = \begin{cases} \sigma_{ij} & \text{for } j \in [(i-1)B, iB] \\ 0 & \text{o/w} \end{cases}$$

Our algorithm simply outputs $\|SA\|_{q\to p}$. The proof of the theorem below can be found in the full version's Appendix D.

▶ **Theorem 35.** *Given a matrix $A \in \mathbb{R}^{n\times n}$, there exists an $(\tilde{O}(\frac{n^2}{\alpha^2}), \alpha)$-sketching family given by $(\mathcal{S}, f)$ for the $q \to p$ norm for $p \in [1, 2]$.*

6 Further Directions

One interesting direction is to study the low-rank approximation problem with respect to the $q \to p$ norm. An important open question in the literature is to find input sparsity time low rank approximation algorithms with respect to the $2 \to 2$ norm, and a natural step might be to try this problem with for $q \to p$ norms for certain q and p.

Another interesting problem would be to investigate algorithms for approximate nearest neighbors with respect to the $q \to p$ norm, in light of a question posed by [8] about what metric spaces admit efficient approximate nearest neighbor algorithms, with matrix norms mentioned as an object of interest.

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 Theory of computing*, pages 20–29. ACM, 1996.

2 Alexandr Andoni. Nearest neighbor search in high-dimensional spaces. In *the workshop: Barriers in Computational Complexity II*, 2010. URL: `http://www.mit.edu/~andoni/nns-barriers.pdf`.

3 Alexandr Andoni. High frequency moments via max-stability. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017, New Orleans, LA, USA, March 5-9, 2017*, pages 6364–6368, 2017.

4 Alexandr Andoni et al. Eigenvalues of a matrix in the streaming model. In *Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algorithms*, pages 1729–1737. Society for Industrial and Applied Mathematics, 2013.

5 Alexandr Andoni, T. S. Jayram, and Mihai Patrascu. Lower bounds for edit distance and product metrics via poincaré-type inequalities. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, pages 184–192, 2010.

6 Alexandr Andoni, Robert Krauthgamer, and Krzysztof Onak. Streaming algorithms via precision sampling. In *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*, pages 363–372. IEEE, 2011.

7 Alexandr Andoni, Robert Krauthgamer, and Ilya P. Razenshteyn. Sketching and embedding are equivalent for norms. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 479–488, 2015.

8 Alexandr Andoni, Huy L Nguyen, Aleksandar Nikolov, Ilya Razenshteyn, and Erik Waingarten. Approximate near neighbors for general symmetric norms. In *Proceedings*

of the 49th Annual ACM SIGACT Symposium on Theory of Computing, pages 902–913. ACM, 2017.

9 Alexandr Andoni, Huy L Nguyên, Yury Polyanskiy, and Yihong Wu. Tight lower bound for linear sketches of moments. In *International Colloquium on Automata, Languages, and Programming*, pages 25–32. Springer, 2013.

10 Ziv Bar-Yossef, Thathachar S Jayram, Ravi Kumar, and D Sivakumar. An information statistics approach to data stream and communication complexity. In *Foundations of Computer Science, 2002. Proceedings. The 43rd Annual IEEE Symposium on*, pages 209–218. IEEE, 2002.

11 Boaz Barak, Fernando GSL Brandao, Aram W Harrow, Jonathan Kelner, David Steurer, and Yuan Zhou. Hypercontractivity, sum-of-squares proofs, and their applications. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 307–326. ACM, 2012.

12 Aditya Bhaskara and Aravindan Vijayaraghavan. Approximating matrix p-norms. In *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms*, pages 497–511. SIAM, 2011.

13 Jaroslaw Blasiok, Vladimir Braverman, Stephen R. Chestnut, Robert Krauthgamer, and Lin F. Yang. Streaming symmetric norms via measure concentration. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 716–729, 2017.

14 Fernando GSL Brandão and Aram W Harrow. Estimating operator norms using covering nets. *arXiv preprint arXiv:1509.05065*, 2015.

15 V. Braverman, S. R. Chestnut, R. Krauthgamer, Y. Li, D. P. Woodruff, and L. F. Yang. Matrix Norms in Data Streams: Faster, Multi-Pass and Row-Order. *ArXiv e-prints*, 2016. `arXiv:1609.05885`.

16 Jop Briët, Fernando Mário de Oliveira Filho, and Frank Vallentin. The positive semidefinite grothendieck problem with rank constraint. In *Automata, Languages and Programming, 37th International Colloquium, ICALP 2010, Bordeaux, France, July 6-10, 2010, Proceedings, Part I*, pages 31–42, 2010.

17 Jop Briët, Oded Regev, and Rishi Saket. Tight hardness of the non-commutative grothendieck problem. *Theory of Computing*, 13(1):1–24, 2017.

18 Kenneth L Clarkson and David P Woodruff. Low rank approximation and regression in input sparsity time. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 81–90. ACM, 2013.

19 Graham Cormode and S Muthukrishnan. Space efficient mining of multigraph streams. In *Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 271–282. ACM, 2005.

20 Uffe Haagerup. The best constants in the khintchine inequality. *Studia Mathematica*, 70(3):231–283, 1981.

21 Moritz Hardt and Eric Price. Tight bounds for learning a mixture of two gaussians. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 753–760. ACM, 2015.

22 Aram W Harrow, Ashley Montanaro, and Anthony J Short. Limitations on quantum dimensionality reduction. In *International Colloquium on Automata, Languages, and Programming*, pages 86–97. Springer, 2011.

23 Piotr Indyk. Stable distributions, pseudorandom generators, embeddings, and data stream computation. *Journal of the ACM (JACM)*, 53(3):307–323, 2006.

24 Piotr Indyk and David Woodruff. Optimal approximations of the frequency moments of data streams. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 202–208. ACM, 2005.

25 T. S. Jayram. On the information complexity of cascaded norms with small domains. In *2013 IEEE Information Theory Workshop, ITW 2013, Sevilla, Spain, September 9-13, 2013*, pages 1–5, 2013.

26 Thathachar S Jayram and David P Woodruff. The data stream space complexity of cascaded norms. In *Foundations of Computer Science, 2009. FOCS'09. 50th Annual IEEE Symposium on*, pages 765–774. IEEE, 2009.

27 Daniel M Kane, Jelani Nelson, Ely Porat, and David P Woodruff. Fast moment estimation in data streams in optimal space. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 745–754. ACM, 2011.

28 Daniel M Kane, Jelani Nelson, and David P Woodruff. On the exact space complexity of sketching and streaming small norms. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 1161–1178. SIAM, 2010.

29 Ashish Khetan and Sewoong Oh. Matrix norm estimation from a few entries. 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 6427–6436, 2017.

30 Subhash A Khot and Nisheeth K Vishnoi. The unique games conjecture, integrality gap for cut problems and embeddability of negative-type metrics into ℓ_1. *Journal of the ACM (JACM)*, 62(1):8, 2015.

31 Weihao Kong and Gregory Valiant. Spectrum estimation from samples. *CoRR*, abs/1602.00061, 2016.

32 Yi Li, Huy L Nguyên, and David P Woodruff. On sketching matrix norms and the top singular vector. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 1562–1581. Society for Industrial and Applied Mathematics, 2014.

33 Yi Li and David P. Woodruff. A tight lower bound for high frequency moment estimation with small error. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques - 16th International Workshop, APPROX 2013, and 17th International Workshop, RANDOM 2013, Berkeley, CA, USA, August 21-23, 2013. Proceedings*, pages 623–638, 2013.

34 Yi Li and David P. Woodruff. On approximating functions of the singular values in a stream. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 726–739, 2016.

35 Yi Li and David P Woodruff. Tight bounds for sketching the operator norm, schatten norms, and subspace embeddings. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 60. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.

36 Yi Li and David P. Woodruff. Embeddings of schatten norms with applications to data streams. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, pages 60:1–60:14, 2017.

37 Jirı Matoušek. Lecture notes on metric embeddings. Technical report, Technical report, ETH Zürich, 2013.

38 Cameron Musco, Praneeth Netrapalli, Aaron Sidford, Shashanka Ubaru, and David P. Woodruff. Spectrum approximation beyond fast matrix multiplication: Algorithms and hardness. In *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, pages 8:1–8:21, 2018.

39 Assaf Naor, Oded Regev, and Thomas Vidick. Efficient rounding for the noncommutative grothendieck inequality. *Theory of Computing*, 10:257–295, 2014.

40 Eric Price and David P. Woodruff. Applications of the shannon-hartley theorem to data streams and sparse recovery. In *Proceedings of the 2012 IEEE International Symposium on Information Theory, ISIT 2012, Cambridge, MA, USA, July 1-6, 2012*, pages 2446–2450, 2012.

41 Tamas Sarlos. Improved approximation algorithms for large matrices via random projections. In *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*, pages 143–152. IEEE, 2006.

42 Zhao Song, David P. Woodruff, and Peilin Zhong. Low rank approximation with entrywise l_1-norm error. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 688–701, 2017.

43 Terence Tao. *Topics in random matrix theory*, volume 132. American Mathematical Society Providence, RI, 2012.

44 Shashanka Ubaru, Jie Chen, and Yousef Saad. Fast estimation of tr(f(a)) via stochastic lanczos quadrature, 2016. URL: `http://www-users.cs.umn.edu/~saad/PDF/ys-2016-04.pdf`.

45 Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.

46 Andreas J. Winter. Quantum and classical message identification via quantum channels. *Quantum Information & Computation*, 5(7):605–606, 2005.

47 David P. Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends in Theoretical Computer Science*, 10(1-2):1–157, 2014.

A Proofs from Section 2

Proof of Lemma 9. For any x that is unit according to ℓ_1,

$$\begin{aligned}\|Ax\|_p &= \|A_{*,1}x_1 + A_{*,2}x_2 + \ldots + A_{*,n}x_n\|_p \\ &\leq \|A_{*,1}\|_p|x_1| + \|A_{*,2}\|_p|x_2| + \ldots + \|A_{*,n}\|_p|x_n| \leq \max_{i\in[n]}\{\|A_{*,i}\|_p\}\end{aligned}$$

where the last inequality is because $|x_i|$ give a convex combination and is achieved for $x = e_{i^*}$ where $i^* = \arg\max_i\{\|A_{*,i}\|_p\}$. ◀

Proof of Lemma 10. For any x such that there is a coordinate x_j that is strictly between 1 or -1, let ε be $\min\{1 - x_j, x_j + 1\}$, consider

$$\begin{aligned}\|Ax\|_p &= \|A_{*,j}x_j + \sum_{i\neq j} A_{*,i}x_i\|_p \\ &\leq \left(\frac{1+x_j}{2}\right)\|A_{*,j} + \sum_{i\neq j} A_{*,i}x_i\|_p + \left(\frac{1-x_j}{2}\right)\|-A_{*,j} + \sum_{i\neq j} A_{*,i}x_i\|_p\end{aligned}$$

where the inequality is due to the triangle inequality. Since $\|Ax\|_p$ is at most a convex combination of the p-norms after replacing x_j with 1 or -1, we can make x_j one of 1 or -1 without decreasing the p-norm. ◀

Proof of Lemma 12. Pick x^* on the unit ball such that $\|Ax^*\|_{\mathcal{Y}} = \|A\|_{\mathcal{X}\to\mathcal{Y}}$. There is $x \in N$ such that $\|x^* - x\|_{\mathcal{X}} < \varepsilon$, which means

$$\|A(x^* - x)\|_{\mathcal{Y}} \leq \|A\|_{\mathcal{X}\to\mathcal{Y}}\|x - x^*\|_{\mathcal{X}} < \varepsilon\|A\|_{\mathcal{X}\to\mathcal{Y}}$$

On the other hand,

$$\|A(x^* - x)\|_{\mathcal{Y}} \geq \|Ax^*\|_{\mathcal{Y}} - \|Ax\|_{\mathcal{Y}} \geq \|A\|_{\mathcal{X}\to\mathcal{Y}} - \|Ax\|_{\mathcal{Y}}$$

and hence

$$\begin{aligned}\|A\|_{\mathcal{X}\to\mathcal{Y}} - \|Ax\|_{\mathcal{Y}} &< \varepsilon\|A\|_{\mathcal{X}\to\mathcal{Y}} \\ \|A\|_{\mathcal{X}\to\mathcal{Y}} < \frac{\|Ax\|_{\mathcal{Y}}}{1-\varepsilon} &\leq \frac{1}{1-\varepsilon}\max_{x\in N}\|Ax\|_{\mathcal{Y}}\end{aligned}$$ ◀

Proof of Lemma 13. For x in a normed space $\mathcal{X}$, we use the notation $B_x(r)$ to denote $\{y : \|x-y\|_{\mathcal{X}} < r\}$, the ball of radius r around x.

Start with an empty set N and while there is a point x in the unit ball B that has distance at least ε to every element in N, pick x and add it to N. This process terminates when every $x \in B$ has distance less than ε to some element in N, thereby terminating with N as an ε-net. We claim that the size of N meets the desired bound.

By construction, any y and y' in N are at least ε apart, which means $\mathcal{B} = \{B_x(\varepsilon/2) : x \in N\}$ is a collection of disjoint sets and note that

$$\bigcup_{S\in\mathcal{B}} S \subseteq B_0(1+\varepsilon/2)$$

By disjointness

$$\mathrm{Vol}\left(\bigcup_{S\in\mathcal{B}} S\right) = \sum_{S\in\mathcal{B}} \mathrm{Vol}(S) = |N|\mathrm{Vol}(B_0(\varepsilon/2))$$

where $\mathrm{Vol}(S)$ is the volume of S according to the Lebesgue measure.

And thus, we obtain

$$\begin{aligned}
|N| &= \frac{\mathrm{Vol}\left(\bigcup_{S\in\mathcal{B}} S\right)}{\mathrm{Vol}(B_0(\varepsilon/2))} \\
&\leq \frac{\mathrm{Vol}(B_0(1+\varepsilon/2))}{\mathrm{Vol}(B_0(\varepsilon/2))} \\
&= \left(\frac{1+\varepsilon/2}{\varepsilon/2}\right)^n \\
&= \left(\frac{2+\varepsilon}{\varepsilon}\right)^n
\end{aligned}$$

which concludes the proof. ◀

B Missing proofs from Section 3

Proof of Lemma 16. Draw $c\log n$ matrices $S_1, S_2, \ldots, S_{c\log n}$ from $\mathcal{D}$ independently where c is a constant to be determined later. We define

$$S := \begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_{c\log n} \end{bmatrix}$$

$$g(Sx) := \mathsf{median}\{f(S_1x), f(S_2x), \ldots, f(S_{c\log n}x)\}$$

Let's analyze the probability that $g(Sx)$ falls outside $L_x = \left(\frac{1}{2}\|x\|_p, 2\|x\|_p\right)$. In order for that to happen, more than half of $f(S_1x), \ldots, f(S_{c\log n}x)$ must lie outside L_x, and this happens to each $f(S_ix)$ with probability at most $\frac{1}{3}$. Using Hoeffding's inequality, we know

$$\mathbf{Pr}[g(Sx) \notin L] \leq 2\exp\left(-\frac{c\log n}{72}\right)$$

which for appropriate choice of c can be bounded by $\frac{1}{n^2}$.

For a matrix A with n columns, a union bound tells us that for all i, $g(SA_{*,i})$ falls in $L_{A_{*,i}}$ with probability at least $1-\frac{1}{n}$. Combined with Lemma 3.1, it follows that $h(SA) := \max_i g(SA_{*,i})$ is a 2-approximation to $\|A\|_{1\to p}$ with probability at least $1-\frac{1}{n}$. ◀

C Missing Proofs from Section 4

C.1 Missing Proofs from Section 4.3

Proof of Lemma 25. We denote $C\kappa$ as α and set the exact value of α in the end of the proof. For a fixed pair i, j let us denote the perturbation term $\alpha\eta_p e_i e_j^\top$ as E_{ij}. Recall that from section 3.1, we know that $\|A\|_{1\to p} = \max_{i\in[n]} \|A_{*,i}\|_p$ which means that it suffices to give bounds on the maximum ℓ_p norm across columns of G_1 and G_2 respectively.

Since the ℓ_p norm is 1-Lipschitz for any $p \geq 2$, we can apply Theorem 6 to show concentration around the expectation for $\|G_{*,i}\|_p$ for any column i of a matrix G of i.i.d Gaussian entries. Hence we have that for any column i, and some positive constant λ

$$\mathbf{Pr}\left[\|G_{*,i}\|_p \geq \lambda \mathbf{E}[\|G_{*,i}\|_p]\right] \leq C\exp(-c\lambda^2 \mathbf{E}[\|G_{*,i}\|_p]^2)$$

Letting g be an n-dimensional vector of i.i.d Gaussians, since we know $\mathbf{E}[\|g\|_p] = \Omega(\sqrt{\log n})$, there exists appropriate constant β such that for any column i of G_1 we have that $\|(G_1)_{*,i}\|_p$ is less than $\beta\mathbf{E}[\|g\|_p]$ with probability at least $1 - \frac{1}{n^2}$. By a union bound over all columns, the probability that $\|G_1\|_{1\to p} \leq \beta\mathbf{E}[\|g\|_p]$ is at least $1 - \frac{1}{n}$.

For a matrix $G_2 = G + E_{ij}$ drawn from $\mathcal{G}_2[\alpha, \eta_p]$, we know that the perturbed column j has norm at least $\alpha\eta_p - \|G_{*,i}\|_p$, which satisfies $(\alpha - \beta)\mathbf{E}[\|g\|_p] \leq \|G_2\|_{1\to p}$. Setting $\alpha \geq (\kappa + 1)\beta$ gives us the desired result. ◀

Proof of Lemma 26. Recall perturbation term $\alpha\eta_p e_i e_j^\top$ was referred to as E_{ij}. Just as in Lemma 22, we can think of L as a $k \times n^2$ matrix that acts on a sample from $\mathcal{G}_1$ or $\mathcal{G}_2[\alpha]$ as though it were an n^2-dimensional vector. Without loss of generality, we can assume that the rows of L are orthonormal, since as before we can always perform a change of basis in post-processing. Thus, the distribution $\mathcal{D}_1$ is the same as $\mathcal{N}(0, I_k)$. For fixed i, j, the distribution of $L(G + E_{ij})$ is Gaussian with mean vector $L(E_{ij})$ (the ij^{th} column of the $k \times n^2$ matrix L scaled by $\alpha\eta_p$) and covariance I_k because of the following.

$$\begin{aligned}
\mathsf{Cov}(L(G + E_{ij})) &= \mathbf{E}\left[\left(L(G + E_{ij}) - \mathbf{E}\left[L(G + E_{ij})\right]\right)^\top \left(L(G + E_{ij}) - \mathbf{E}\left[L(G + E_{ij})\right]\right)\right] \\
&= \mathbf{E}\left[\left(L(G) - \mathbf{E}\left[L(G)\right]\right)^\top \left(L(G) - \mathbf{E}\left[L(G)\right]\right)\right] \\
&= \mathsf{Cov}_{G\sim\mathcal{N}(0,I_n)}(G) = I_k
\end{aligned}$$

Thus, $\mathcal{D}_2$ is the distribution of picking a random i, j and drawing a matrix from $\mathcal{N}(L(E_{ij}), I_k)$.

We now analyze the total variation distance between $\mathcal{D}_1$ and $\mathcal{D}_2$ and get the desired bound from a chain of inequalities.

$$\begin{aligned}
d_{TV}(\mathcal{D}_1, \mathcal{D}_2) &= \frac{1}{2}\int_{x\in\mathbb{R}^k} |p_{\mathcal{D}_1}(x) - p_{\mathcal{D}_2}(x)|dx \\
&= \frac{1}{2}\int_{x\in\mathbb{R}^k} \left|\sum_{i,j} \frac{1}{n^2} p_{\mathcal{D}_1}(x) - \frac{1}{n^2} p_{\mathcal{N}(L(E_{ij}),I_k)}(x)\right| dx \\
&\leq \frac{1}{n^2}\sum_{i,j} \frac{1}{2}\int_{x\in\mathbb{R}^k} \left|p_{\mathcal{D}_1}(x) - p_{\mathcal{N}(L(E_{ij}),I_k)}\right| dx \\
&= \frac{1}{n^2}\sum_{i,j} d_{TV}(\mathcal{D}_1, \mathcal{N}(L(E_{ij}), I_k))
\end{aligned}$$

$$
\begin{aligned}
&= \frac{1}{n^2}\sum_{i,j} d_{TV}(\mathcal{N}(0,I_k),\mathcal{N}(L(E_{ij}),I_k)) \\
&\le \frac{1}{n^2}\sum_{i,j} C'\alpha\eta_p\|L_{*,ij}\|_2 && \text{[from lemma 5]} \\
&= \frac{C'\alpha\eta_p}{n^2}\|L\|_{1,2} \\
&\le \frac{C'\alpha\eta_p}{n^2}\cdot n\|L\|_F = C'\alpha\eta_p\cdot\frac{\sqrt{k}}{n} && \text{[by Cauchy-Schwarz]}
\end{aligned}
$$

◀

C.2 Missing Proofs from Section 4.6

Proof of Lemma 32. We claim that for a diagonal matrix D, $\arg\max_{\|x\|_q=1}\|Dx\|_p$ is achieved when x is one of the e_i standard basis vectors e_i. To see this,

$$\|Dx\|_p^p = \sum_{i=1}^n |d_{ii}x_i|^p = \sum_{i=1}^n |d_{ii}|^p(|x_i|^q)^{p/q} \le \sum_{i=1}^n |d_{ii}|^p|x_i|^q \le \max_i |d_{ii}|^p$$

which is achieved by picking $x = e_{i^*}$ where choice of $i = i^*$ maximizes d_{ii}.

Thus, to analyze the $q \to p$ norm of G_1, it suffices to analyze $\max_{x\in\{e_i\}}\|G_1x\|_p$, which is the same as $\|g\|_\infty$ where g is a vector of i.i.d. Gaussians. We can extract from the proof of Lemma 25 that $\|g\|_\infty$ is upper bounded by $\beta\sqrt{\log n}$ with probability at least $1-\frac{1}{n^2}$.

On the other hand, if the perturbation is at index (i,i) and we pick $\alpha = \kappa(\beta+1)$, then $\|G_2e_i\|_p$ is at least $\kappa\beta\sqrt{\log n}$ with probability at least $1-\frac{1}{n^2}$ implying the desired separation. ◀

D General approximation factors α

D.1 Sketching Matrix Construction and Upper Bounds

Let us first define our sketch and then analyze its performance. For the sketch S, we group the rows of A into $\frac{n}{\alpha^2}$ groups of size α^2. We label the groups by $B_1,\ldots,B_{n/\alpha^2}$ and let $\sigma_{1i},\ldots,\sigma_{\alpha^2 i}$ be ±1 i.i.d random variables with equal probability for block B_i. Notice then that the i^{th} row of SA given by $(SA)_{i,*}$ is:

$$(SA)_{i,*} \triangleq \sum_{j\in B_i}\sigma_{ji}A_{i,*}$$

To analyze the performance of this sketch, we will need a helpful inequality describing the behavior of a random signed sums of reals.

▶ **Theorem 36** (Khintchine's Inequality, [20]). *Let $\{x_i\}_{i=1}^n \in \mathbb{R}$ be reals and let $\{\mathbf{s_i}\}_{\mathbf{i=1}}^{\mathbf{n}}$ be i.i.d ±1 random variables with equal probability and let $0 < t < \infty$, we then have:*

$$A_p\sqrt{\sum_{i=1}^n x_i^2} \le \boldsymbol{E}\left[\left|\sum_{i=1}^n \mathbf{s_i}\mathbf{x_i}\right|^p\right]^{1/p} \le B_p\sqrt{\sum_{i=1}^n x_i^2}$$

For some constants A_p, B_p that only depend on p.

Also recall that by Jensen's inequality, we can relate two norms of a vector $x \in \mathbb{R}^n$.

▶ Remark. For two positive reals, $p \geq q > 1$ and for a vector $x \in \mathbb{R}^n$ we have that: $\|x\|_p \leq n^{\frac{1}{q}-\frac{1}{p}} \|x\|_q$

We then have the following theorems describing the sketching complexity of the sketch S for $1 \leq p \leq 2$ and for $p > 2$.

▶ **Theorem 37.** *For any $1 \leq p \leq 2$ and for the maximizer $x \in \mathbb{R}^n$ of $\|A\|_{q\to p}$ the sketch S defined earlier where each block B_i has size B has the property that*

$$\Theta(1)\frac{1}{B^{1-\frac{1}{p}}} \|SAx\|_p \leq \|Ax\|_p \leq \Theta(1)B^{\frac{1}{p}-\frac{1}{2}} \|SAx\|_p$$

with probability at least $\frac{99}{100}$

Proof. Let us first show the first inequality in the theorem statement.

For some coordinate $1 \leq i \leq \frac{n}{B}$:

$$|(SAx)_i|^p = \left|\sum_{j\in B_i} \sigma_j (Ax)_j\right|^p \leq \left(\sum_{j\in B_i} |(Ax)_j|\right)^p$$

By Remark D.1 relating $\|\cdot\|_1$ and $\|\cdot\|_p$

$$\leq B^{p-1} \sum_{j\in B_i} |(Ax)_j|^p$$

$$\therefore \|(SAx)_i\|_p = \left(\sum_{i=1}^{n/B} |(SAx)_i|^p\right)^{1/p} \leq B^{1-\frac{1}{p}} \|Ax\|_p$$

Notice that the first inequality holds irrespective of the vector x, it holds for all vectors. Now let us show the second inequality of the theorem statement.

For some coordinate $1 \leq i \leq \frac{n}{B}$:

$$\left(\sum_{j\in B_i} (Ax)_j^p\right)^{1/p} \leq B^{\frac{1}{p}-\frac{1}{2}} \left(\sum_{j\in B_i} (Ax)_j^2\right)^{1/2} \quad \text{[By Remark D.1]} \quad [1]$$

$$\leq \Theta(1)B^{\frac{1}{p}-\frac{1}{2}} \mathbf{E}\left[\left|\sum_{j\in B_i} \sigma_j (Ax)_j\right|^p\right]^{1/p} \quad \text{[By Khintchine's Ineq.]} \quad [2]$$

$$\therefore \sum_{i=1}^{n/B} \sum_{j\in B_i} (Ax)_j^p = \|Ax\|_p^p \leq \Theta(1)B^{p\left(\frac{1}{p}-\frac{1}{2}\right)} \mathbf{E}\left[\|SAx\|_p^p\right]$$

Notice that the second inequality of the theorem statement follows by Markov's inequality.

Notice that the success probability of line [2] is constant for each block. To get constant success probability over the entire set of blocks, we construct $O(\log(n))$ i.i.d copies of each block B_i given by $\{B_i^j\}_{i=1}^{O(\log(n))}$. We then pick j such that it is the index realizing the quantity $\mathsf{median}_{j\in[O(\log(n))]} \|(S_j Ax)_i\|_p$ where S_j corresponds the sketch with the j^{th} copy of the blocks. Then, by standard concentration bounds, we can get $1 - \frac{1}{n/B}$ success probability for each set of blocks B_i and then union bound over the $\frac{n}{B}$ blocks giving us constant success probability. ◀

▶ **Theorem 38.** *For any $p > 2$ and for the maximizer $x \in \mathbb{R}^n$ of $\|A\|_{q\to p}$ the sketch S defined earlier where each block B_i has size B has the property that*

$$\Theta(1)\frac{1}{B^{1-\frac{1}{p}}}\|SAx\|_p \leq \|Ax\|_p \leq \Theta(1)\|SAx\|_p$$

The proof for Theorem 38 is the same as that for Theorem 37 except that there is no dilation while upper bounding the $\|Ax\|_p$ with the 2-norm in line [1] of the proof.

Notice that the above theorems imply that the sketch S is a $\sqrt{B}$-approximation when $0 \leq p \leq 2$ and a $B^{1-\frac{1}{p}}$-approximation when $p > 2$ because it states that the sketch is stretching $\|Ax\|_p^p$ by at most some factor and dilating it by at most some factor and hence the approximation ratio is simply the product of these factors.

Flow-time Optimization for Concurrent Open-Shop and Precedence Constrained Scheduling Models

Janardhan Kulkarni
Microsoft Research, Redmond, WA, USA
jakul@microsoft.com

Shi Li[1]
Department of Computer Science and Engineering, University at Buffalo, Buffalo, NY, USA
shil@buffalo.edu

Abstract

Scheduling a set of jobs over a collection of machines is a fundamental problem that needs to be solved millions of times a day in various computing platforms: in operating systems, in large data clusters, and in data centers. Along with makespan, *flow-time*, which measures the length of time a job spends in a system before it completes, is arguably the most important metric to measure the performance of a scheduling algorithm. In recent years, there has been a remarkable progress in understanding flow-time based objective functions in diverse settings such as unrelated machines scheduling, broadcast scheduling, multi-dimensional scheduling, to name a few.

Yet, our understanding of the flow-time objective is limited mostly to the scenarios where jobs have no dependencies. On the other hand, in almost all real world applications, think of MapReduce settings for example, jobs have dependencies that need to be respected while making scheduling decisions. In this paper, we take first steps towards understanding this complex problem. In particular, we consider two classical scheduling problems that capture dependencies across jobs: 1) *concurrent open-shop scheduling* (COSSP) and 2) *precedence constrained scheduling*. Our main motivation to study these problems specifically comes from their relevance to two scheduling problems that have gained importance in the context of data centers: *co-flow scheduling* and *DAG scheduling*. We design almost optimal approximation algorithms for COSSP and PCSP, and show hardness results.

2012 ACM Subject Classification Theory of computation → Scheduling algorithms

Keywords and phrases Approximation, Weighted Flow Time, Concurrent Open Shop, Precedence Constraints

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.16

Related Version A full version of the paper is available at `https://arxiv.org/abs/1807.02553`.

1 Introduction

Scheduling a set of jobs over a collection of machines is a fundamental problem that needs to be solved millions of times a day in various computing platforms: in operating systems, in large data clusters, and in data centers. Along with makespan, *flow-time*, which measures the length of time a job spends in a system before completing, is arguably the most important

[1] The work of the author is in part supported by NSF grants CCF-1566356 and CCF-1717134.

metric to measure the performance of a scheduling algorithm. In recent years, there has been a remarkable progress in understanding flow-time related objective functions in diverse settings such as unrelated machines scheduling [22, 31, 5, 29, 10], broadcast scheduling [7, 32, 9], multi-dimensional scheduling [28, 27], to name a few.

Yet, our understanding of the flow-time based objective functions is mostly limited to the scenarios where jobs have simple structures; in particular, each job is a single self contained entity. On the other hand, in almost all real world applications, jobs have more complex structures. Consider the MapReduce model for example. Here, each job consists of a set of *Map* tasks and a set of *Reduce* tasks. Reduce tasks cannot be processed unless Map tasks are completely processed[2]. A MapReduce job is complete only when all Map and Reduce tasks are completed. Motivated by these considerations, in this paper, we consider two classical scheduling models that capture more complex job structures: 1) *concurrent open-shop scheduling (*COSSP*)*, and 2) *precedence constrained scheduling (*PCSP*)*. Our main reason to study these problems specifically comes from their relevance to two scheduling problems that have gained importance in the context of data centers: *co-flow scheduling and DAG scheduling*. We discuss more about how these problems relate to COSSP and PCSP in Section 1.3.

The objective function we consider in this paper is minimizing the *sum of general delay costs of jobs*, first introduced in an influential paper by Bansal and Pruhs [11] in the context of single machine scheduling. In this objective, for each job j we are given a *non-decreasing* function $g_j : \mathbb{Z}_+ \to \mathbb{Z}_+$, which gives the cost of completing the job at time t. The goal is to minimize $\sum_j g_j(C_j)$, where C_j is the completion time of j. A desirable aspect of the general delay cost functions is that they capture several widely studied flow-time and completion time based objective functions.

- Minimizing the sum of weighted flow-times of jobs. This is captured by the function $g_j(t) = w_j \cdot (t - r_j)$, where r_j is the release time of j.
- Minimizing the sum of weighted pth power of flow-times of jobs. This is captured by the function $g_j(t) = w_j \cdot (t - r_j)^p$.
- Minimizing the sum of weighted tardiness. This is captured by the function $g_j(t) = w_j \cdot \max\{0, (t - d_j)\}$, where d_j is the deadline of j.

In this paper, we design approximation algorithms for minimizing the sum of general delay costs of jobs for the concurrent open-shop scheduling and the precedence constrained scheduling problems.

1.1 Concurrent Open-shop Scheduling Problem (COSSP)

In COSSP, we are given a set of m machines and a set of n jobs. Each job j has a release time r_j. The main feature of COSSP is that a job consists of m operations $O_{1j}, O_{2j}, \ldots O_{mj}$, one for each machine $i \in [m]$. Each operation O_{ij} needs p_{ij} units of processing on machine i. We allow operations to have *zero processing lengths*. Throughout the paper, we assume without loss of generality that all our input parameters are positive integers. A job is complete only when *all its operations are complete*. That is, if C_j denotes the completion time of j, then p_{ij} units of operation O_{ij} must be processed in the interval $[r_j, C_j]$ on machine i. In the concurrent open-shop scheduling model, multiple operations of the same job *can be processed simultaneously* across different machines.

[2] In some MapReduce applications, Reduce tasks can begin after the completion of a subset of Map tasks.

The problem has a long history due to its applications in manufacturing, automobile and airplane maintenance and repair [53], etc., and has been extensively studied both in operations research and approximation algorithm communities [3, 16, 23, 43, 52, 51, 39, 8]. As minimizing makespan in COSSP model is trivial, much of the research has focused on the objective of minimizing the total weighted completion times of jobs. The problem was first considered by Ahmadi and Bagchi [3], who showed the NP-hardness of the problem. Later, several groups of authors, Chen and Hall [16], Garg et al. [23], Leung et al. [39], and Mastrolilli et al. [43], designed 2-approximation algorithms for the problem. Under Unique Games Conjecture, Bansal and Khot showed that this approximation factor cannot be improved for the problem [8].

Garg, Kumar, and Pandit [23] studied the more difficult objective of minimizing the total flow-time of jobs, and showed that the problem cannot be approximated better than $\Omega(\log m)$, by giving a reduction from the set cover problem. However, they did not give any approximation algorithm to the problem, and left it as an open problem. To the best of our knowledge, the problem has remained open ever since. In this paper, we make progress on this problem.

Let P denote the ratio of the maximum processing length among all the operations to the minimum non-zero processing length of among all the operations; that is, $P := \frac{\max_{i,j}\{p_{ij}\}}{\min_{i,j}\{p_{ij}:p_{ij}\neq 0\}}$.

▶ **Theorem 1.** *For the objective of minimizing the sum of general delay cost functions of jobs in the concurrent open-shop scheduling model, there exists a polynomial time $O(\log(m \log P))$ approximation algorithm.*

We obtain the above result by generalizing the algorithm in Bansal and Pruhs [11]. Note that when $m = 1$, our result gives a $O(\log \log P)$ approximation algorithm to the problem, matching the best known polynomial time result in [11]. Recently, for the special case of total weighted flow-time, a constant factor approximation algorithm was obtained by Batra, Garg, and Kumar [13] when $m = 1$. However, running time of their algorithm is pseudo-polynomial. Since the approach in [13] is very different from the one in [11], our result does not generalize [13].

As we discussed earlier, the general delay cost functions capture several widely studied performance metrics. Thus, we get:

▶ **Corollary 2.** *There is a polynomial time $O(\log(m \log P))$ approximation algorithm in the concurrent open-shop scheduling model for the following objective functions: 1) Minimizing the sum of weighted flow-times of jobs; 2) Minimizing the weighted ℓ_k-norms of flow-times of jobs; the approximation factor becomes $O((\log(m \log P)^{\frac{1}{k}}))$; 3) Minimizing the sum of weighted tardiness of jobs.*

We give the proof Theorem 1 in Section 2.

1.2 Precedence Constrained Scheduling Problem (PCSP)

More complex forms of job structures are captured by the precedence constrained scheduling problem (PCSP), another problem that has a long history dating back to the seminal work of Graham [24]. Here, we have a set of m identical machines and a set of n jobs; each job j has a processing length $p_j > 0$, a release time $r_j > 0$. Each job j must be scheduled on exactly one of the m machines. The important feature of the problem is that they are precedence constraints between jobs that capture the computational dependencies across jobs. The precedence constraints are given by a partial order "$\prec$", where a constraint $j \prec j'$ requires that job j' can only start after job j is completed. Our goal is to schedule (preemptively) each job on exactly one machine to minimize $\sum_j g_j(C_j)$.

Precedence constrained scheduling on identical machines to minimize the makespan objective is perhaps the most well-known problem in scheduling theory. Already in 1966, Graham showed that list scheduling gives a 2-approximation algorithm to the problem. Since then several attempts have been made to improve the approximation factor [37, 21]. However, Svensson [49] showed that problem does not admit a $2-\epsilon$ approximation under a strong version of the Unique Games Conjecture introduced by Bansal and Khot [8]. An unconditional hardness of $(4/3-\epsilon)$ is also known due to Lenstra and Rinnooy Kan [38]. Recently, Levey and Rothvoss [40] showed that it is possible to overcome these lowerbounds for the case when m is fixed. An LP-hierarchy lift of the time-index LP with a slightly super poly-logarithmic number of rounds provides a $(1+\epsilon)$ approximation to the problem.

Another problem that is extensively studied in the precedence constrained scheduling model is the problem of minimizing the total weighted completion times of jobs. Note that this problem strictly generalizes the makespan problem, hence all the lowerbounds also extend to this problem. The current best approximation factor of 3.387 is achieved by a very recent result of Li [41]. The work builds on a 4-approximation algorithm due to Munier, Queyranne and Schulz ([45], [47]).

In a recent work, Agrawal *et al* [2] initiated the study of minimizing the total flow-time objective in DAG (Directed Acyclic Graphs) *parallelizability* model. In this model, each job is a DAG, and a job completes only when all the nodes in the DAG are completed. For this problem, they showed greedy online algorithms that are constant competitive when given $(1+\epsilon)$-speed augmentation. The DAG parallelizability model is a special case of PCSP. However, as there are no dependencies between jobs, and individual nodes of the DAG do not contribute to the total flow-time unlike in PCSP, complexity of the problem is significantly different from PCSP. For example, when there is only one machine, the DAG structure of individual jobs does not change the cost of the optimal solution, and hence the problem reduces to the standard single machine scheduling problem. Therefore, scheduling jobs using Shortest Remaining Processing Time (SRPT), where processing length of a DAG is its total work across all its nodes, is an optimal algorithm. (Within a DAG, the nodes can be processed in any order respecting the precedence constraints.)

On the other hand, we show a somewhat surprising result for the PCSP problem. We show that the problem of minimizing the total flow-times of jobs does not admit any reasonable approximation factor even on a *single machine*. This is in sharp contrast to makespan and the sum of weighted completion times objective functions that admit $O(1)$-approximation algorithms even on multiple machines. Our hardness proof is based on a recent breakthrough work of Manurangsi [42] on approximating the Densest-k-Subgraph (DkS) problem.

▶ **Theorem 3.** *In the precedence constrained scheduling model, for the objective of minimizing the total flow-times of jobs on a single machine, no polynomial time algorithm can achieve an approximation factor better than* $n^{\frac{1}{(\log\log n)^c}}$*, for some universal constant* $c>0$*, assuming the exponential time hypothesis (ETH).*

To circumvent this hardness result, we study the problem in the speed augmentation model, which can also be thought of as a bi-criteria analysis. In the speed augmentation model, each machine is given some small extra speed compared to the optimal solution. The speed augmentation model was introduced in the seminal work of Kalyanasundaram and Pruhs [34] to analyze the effectiveness of various scheduling heuristics in the context of online algorithms. However, the model has also been used in the offline setting to overcome strong lowerbounds on the approximability of various scheduling problems; see for example results on non-preemptive flow-time scheduling that use $O(1)$-speed augmentation to obtain $O(1)$-approximation ratio [6, 30].

Our second main result is an $O(1)$-approximation algorithm for the problem in the speed augmentation model. Previously, no results were known for the flow-time related objective functions for PCSP.

▶ **Theorem 4.** *For the objective of minimizing the sum of general delay cost of jobs in the precedence constrained scheduling model on identical machines, there exists a polynomial time $O(1)$-speed $O(1)$ approximation algorithm. Furthermore, the speed augmentation required to achieve an approximation factor better than n^{1-c}, for any $c > 0$, has to be at least the best approximation factor of the makespan minimization problem. The lowerbound on speed augmentation extends to any machine environment, such as related and unrelated machine environments.*

We give the proofs of above theorems in the full version of the paper.

1.3 Applications of COSSP and PCSP in Data Center Scheduling

Besides being fundamental optimization problems, COSSP and PCSP models are very closely related to the scheduling problems that arise in the context of data centers. In particular, COSSP is a special case of the *Coflow Scheduling* problem introduced in a very influential work of Choudary and Stoica [17, 19]. On the other hand, PCSP generalizes the DAG scheduling problem, again a widely studied problem in systems literature. In fact, the DAG scheduling model has been adopted by Yarn, the resource manager of Hadoop [1]. See [26, 25] and references there-in for more details. Due to space constraints, we will defer the detailed discussion of connections among these problems to Appendix A.

1.4 Overview of the Algorithms and Techniques

Both our algorithms are based on rounding linear programming relaxations of the problems. However, individual techniques are quite different, and hence we discuss them separately.

1.4.1 Open-shop Scheduling Problem

Our algorithm for COSSP is based on the geometric view of scheduling developed by Bansal and Pruhs [11] for the problem of minimizing the sum of general delay costs of jobs on a single machine, which is a special case of our problem when $m = 1$. The key observation that leads to this geometric view is that minimizing general delay costs is equivalent to coming-up with a set of *feasible deadlines* for all jobs. Moreover, testing the feasibility of deadlines further boils down to ensuring that for *every* interval of time, the total volume of jobs that have (arrival time, deadline) windows within that interval is not large compared to the length. By a string of nice arguments, the authors show that this deadline scheduling problem can be viewed as a capacitated geometric set cover problem called R2C, which stands for capacitated rectangle covering problem in 2-dimensional space.[3] Further, they argue that an α-approximation algorithm for R2C problem can be used to obtain an α-approximation algorithm for the scheduling problem.

In R2C, we are given a set $\mathcal{P}$ of points in 2 dimensions, where each $p \in \mathcal{P}$ is specified by coordinates (x_p, y_p). Associated with each point $p \in \mathcal{P}$ is a demand $d_p > 0$. We are also

[3] "C" stands for the capacitated version in which rectangles have capacities and points have demands. Later, we shall use "M" for the multi-cover version, where rectangles are uncapacitated (or have capacity 1) and points have different demands. We use "U" for the uncapacitated version, where rectangles are uncapacitated and all points have demand 1.

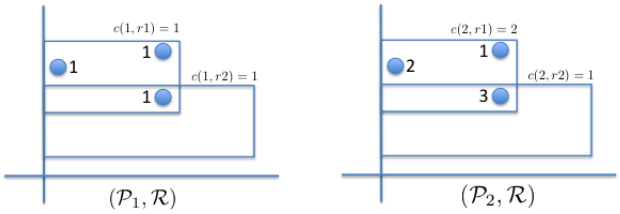


Figure 1 The figure shows an instance of PR2C problem where are there are two sets of points. For this instance the only feasible solution is to pick both the rectangles.

given a set $\mathcal{R}$ of rectangles, where $r \in \mathcal{R}$ has the form $(0, x_r) \times (y_r^1, y_r^2)$. Each rectangle r has a capacity $c(r) > 0$ and a cost $w(r) > 0$. The goal is to choose a minimum-cost set of rectangles, such that for every point $p \in \mathcal{P}$, the total capacity of selected rectangles covering p is at least d_p.

Our problem, which we call PR2C, can be seen as a parallel version of R2C. In PR2C, we have m instances of R2C problem with a common set of rectangles. Namely, the ith instance is defined by $(\mathcal{P}_i, \mathcal{R})$, where each $p \in \mathcal{P}_i$ is associated with a demand d_p, each $r \in \mathcal{R}$ is associated with a capacity $c(i, r)$ and cost $w(r)$. Notice that a rectangle $r \in \mathcal{R}$ has the same cost across the m instances, but has different capacities in different instances. The goal is to find a minimum cost set $X \subseteq \mathcal{R}$ of rectangles that is a valid solution for every R2C instance $(\mathcal{P}_i, \mathcal{R})$. Using the arguments similar to [11], we also show that if there is an α-approximation to PR2C problem, it gives an α-approximation for the COSSP problem.

Thus, much of our work is about designing a good approximation algorithm for the PR2C problem. Our algorithm for PR2C is a natural generalization of the algorithm for R2C of Bansal and Pruhs in [11]. In [11], Bansal and Pruhs formulated an LP relaxation for R2C based on knapsack cover (KC) inequalities. In the LP, $x_r \in [0, 1]$ indicates the fraction of the rectangle r that is selected. If the LP solution picks a rectangle to some constant fraction, then we can also select the rectangle in our solution without increasing the cost by too much. After selecting these rectangles, some points in $\mathcal{P}$ are covered, and some other points p will still have residual demands d'_p. These not-yet-covered points are divided into two categories: *light and heavy points*. Roughly speaking, a point p is heavy if it is mostly covered by rectangles r with $c(r) \geq d'_p$ in the LP solution; a point p is light if it is mostly covered by rectangles r with $c(r) < d'_p$. Heavy points and light points are handled separately. The problem of covering heavy points will be reduced to the R3U problem, a geometric weighted set-cover problem where elements are points in 3D space and sets are axis-parallel cuboids. On the other hand, the problem of covering light points can be reduced to $O(\log_2 P)$ R2M instances. Each R2M instance is a geometric weighted set multi-cover instance in 2D-plane. By appealing to the geometry of the objects produced by the scheduling instance, [11] prove that *union complexity of objects* in R3U and R2M instances is small. In particular, R3U instance has *union complexity* $O(\log P)$ and each R2M instance has union complexity $O(1)$. Using the technique of quasi-uniform sampling introduced in [50, 15] for solving geometric weighted set cover instances with small union complexity, Bansal and Pruhs obtain $O(\log \log P)$ and $O(1)$ approximation ratios for the problems of covering heavy and light points, respectively.

In our problem, we have m parallel R2C instances with a common set of rectangles. As in [11], for each instance, we categorize the points into heavy and light points based on the LP solution. The problem of handling heavy points then can be reduced to m R3U

instances, with the m sets of cuboids identified. However, we cannot solve these m R3U instances separately, because such a solution cannot be mapped back to a valid schedule for COSSP. Therefore, we combine the m instances of R3C into a single instance of a 4 dimensional problem. So, in the combined instance, our geometric objects, which we call hyper-4cuboids, contain (at most) one 4-cuboid (a cuboid in 4 dimensions) from each of the m instances. The goal is to choose a minimum cost set of objects to cover all the points. On the other hand, for the light points, [11] reduced the problem to $O(\log P)$ R2M instances. Again, this is approach is not viable for our case as we need to solve all the instances in parallel. By a simple trick, we first merge the $O(\log P)$ instances into one R2M instance. We then have m R2M instances with a new sets of rectangles identified, which we map into a single 3-dimensional geometric multi-set cover problem.

In both cases we show that the union complexity of the objects in our geometric problems increase at most by a factor of m compared to the objects in [11]. Thus, we have $O(m \log P)$ union complexity for the problem for heavy points and $O(m)$ union complexity for the problem for light points. Using the technique of [11], we obtain $O(\log(m \log P))$ and $O(\log m)$ approximation ratios for heavy and light points respectively, resulting in an $O(\log(m \log P))$ overall approximation ratio.

1.4.2 Precedence Constrained Scheduling

Our algorithm for the precedence constrained scheduling problem works in two steps. In the first step, we construct a *migratory* schedule, in which a job may be processed on multiple machines. For migratory schedules, we can assume that all jobs have unit size by replacing each job j of size p_j with a precedence chain of p_j unit length jobs. Solving a natural LP relaxation for the problem gives us a completion time vector $(C_j)_j$. Then we run the list-scheduling algorithm of Munier, Queyranne and Schulz [45], and Queyranne and Schulz [47], that works for the problem with weighted completion time objective. Specifically, for each job j in non-decreasing order of C_j values, we insert j to the earliest available slot after r_j without violating the precedence constraints.

To analyze the completion time of job j, we focus on the schedule constructed by our algorithm after the insertion of j. A simple lemma is that at any time slot after r_j, we are making progress towards scheduling j in the schedule: either all machines are busy in the time slot, or we are processing a set of jobs whose removal will decrease the "depth" of j in the precedence graph. This lemma was used in [45, 47] to give their 3-approximation for the problem of minimizing weighted completion time (for unit-size jobs), and recently by Li [41] to give an improved 2.415-approximation for the same problem. With 3-speed augmentation, this leads to a schedule that completes every job j by the time C_j. With additional speed augmentation, this leads to an $O(1)$-approximation for the problem with general delay cost functions. In the second step, we convert the migratory schedule into a non-migratory one, using some known techniques (e.g. [35], [20], [33]). The conversion does not increase the completion times of jobs, but requires some extra $O(1)$-speed augmentation.

2 Concurrent Open-shop Scheduling

In this section we consider the concurrent open-shop scheduling problem. Recall that in COSSP, we are given a set of m machines and a set of n jobs. Each job has a release time r_j. A job consists of m operations $\{O_{ij}\}_i$, one for each machine $i \in [m]$. Each operation O_{ij} needs p_{ij} units of processing on machine i. A job finishes only when all its operations are completed. The goal is to construct a preemptive schedule that minimizes the sum of costs incurred by jobs: $\sum_j g_j(C_j)$.

As we mentioned earlier, our algorithm for COSSP is based on the geometric view of scheduling developed in the work of Bansal and Pruhs [11]. Similar to [11], we first reduce our problem to a geometric covering problem that we call Parallel Capacitated Rectangle Covering Problem (PR2C). We argue that an α-approximation to PR2C will give an α approximation to our problem. Then, we design an $O(\log(m \log P))$-factor approximation algorithm to PR2C.

2.1 Reduction to PR2C Problem

In PR2C, we have m instances of R2C problem with a common set of rectangles. Input to the problem consists of sets $\mathcal{P}_1, \mathcal{P}_2, \ldots \mathcal{P}_m$, where each $\mathcal{P}_i$ is a set of points in 2-dimensional space. Each point $p \in \mathcal{P}_i$ is specified by its coordinates (x_p, y_p) and has a demand $d_p > 0$. The input also consists of a set $\mathcal{R}$ of rectangles. Each rectangle $r \in \mathcal{R}$ has the form $(0, x_r) \times (y_r^1, y_r^2)$, and has a cost $w(r)$. Each rectangle r also has a capacity $c(i, r)$ which depends on the point set $\mathcal{P}_i$. Notice that a rectangle $r \in \mathcal{R}$ has the same cost across the m instances, but has different capacities in different instances. The goal is to find a minimum cost set $X \subseteq \mathcal{R}$ of rectangles that is a valid solution for every R2C instance $(\mathcal{P}_i, \mathcal{R})$. Recall that a set of rectangles is a valid solution to an instance of R2C, if for every point the total capacity of rectangles that cover the point is at least the demand of the point.

We can capture the PR2C problem using an integer program. Let x_r be a binary variable that indicates if the rectangle $r \in \mathcal{R}$ is picked in the solution or not. Then, the following integer program IP (1 - 3) captures the PR2C problem.

$$\text{Minimize} \quad \sum_{r \in \mathcal{R}} w(r) x_r \qquad \text{s.t.} \tag{1}$$

$$\forall i, \forall p \in \mathcal{P}_i : \quad \sum_{r \ni p} c(i, r) x_r \geq d_p, \tag{2}$$

$$\forall r \in \mathcal{R} : \quad x_r \in \{0, 1\}. \tag{3}$$

To see the connection between COSSP and PR2C, we need to understand the structure of feasible solutions to COSSP. Consider a feasible schedule S to an instance of COSSP. Suppose the completion time of a job j is C_j in S. This implies that on every machine i, the job j completes its operation O_{ij} in the interval $[r_j, C_j]$. If S is a feasible schedule and $\{C_j\}_j$ is the completion times of jobs, then processing jobs using the Earliest Deadline First (EDF) algorithm on each machine ensures that all jobs finish by their deadline C_j. Thus, one of the main observations behind our reduction is that minimizing the sum of costs incurred by jobs is equivalent to coming up with a deadline C_j for each job j. Thus a natural approach is to formulate COSSP as a deadline scheduling problem. However, this raises the question: Is there a way to test if a set of deadlines $\{C_j\}_j$ is a feasible solution to COSSP? An answer to the question is given by the characterization of when EDF will find a feasible schedule (one where all jobs meet their deadlines) on a single machine. To proceed, we need to set up some notation.

For an interval $I = [t_1, t_2]$, which consists of all the time slots between t_1 and t_2, let $J(I)$ denote the set of jobs that have release times in I: $J(I) = \{j | r_j \in [t_1, t_2]\}$. For a set of jobs J' and a machine i, let $P(i, J')$ denote the total processing length of operations O_{ij} of the jobs in the set J'. Now, we introduce the notion of *excess* for an interval.

▶ **Definition 5.** For a given interval I and a machine i, the excess of I on machine i, denoted by $\xi(i, I)$, is defined as $\max\{0, P(i, J(I)) - |I|\}$.

Following lemma states the condition under which EDF can schedule all the jobs within their deadlines.

▶ **Lemma 6.** *Given a set of jobs with release times and deadlines, scheduling operations on each machine according to* EDF *is feasible if and only if, for every machine i and every interval $I = [t_1, t_2]$, the total processing lengths of operations corresponding to jobs in $J(I)$ that have deadlines greater than t_2 is at least $\xi(i, I)$.*

Clearly, if the total processing lengths of operations corresponding to jobs in $J(I)$ that have deadlines greater than t_2 is less than $\xi(i, I)$, then no algorithm can meet all the deadlines, as the length of operations scheduled in the interval I is greater than $|I|$. Sufficiency of the above lemma follows from a bipartite graph matching argument, and we refer the reader to [11] for more details.

Lemma 6 leads to an integer programming formulation for COSSP, and we shall use this IP to define the PR2C instance. Let $L = \max_i\{\sum_j p_{ij}\}$; note that every reasonable schedule finishes by L. For every job j, and every integer $q \in [-1, \lceil \log_2 g_j(L) \rceil]$, let $t_{j,q}$ be the largest integer $t \in (r_j, L]$ such that $g_j(t) \leq 2^q$ (if such t does not exist, then $t_{j,q} = r_j$). Let $t_{j,-2} = r_j$. For every j and $q \in [-1, \lceil \log_2 g_j(L) \rceil]$, we have a variable $x_{j,q}$ indicating whether $C_j \in (t_{j,q-1}, t_{j,q}]$. Therefore, for a job j, the total number of $x_{j,q}$ variables in our IP will be at most $O(\log g_j(L))$. For every j and every $t > r_j$, let $q_{j,t}$ be the integer q such that $t \in (t_{j,q-1}, t_{j,q}]$.

Our IP for COSSP is as follows:

$$\text{minimize} \quad \sum_j \sum_q \lfloor 2^q \rfloor \, x_{j,q}, \qquad \text{s.t.}$$

$$\forall i, \forall I = [t_1, t_2] : \quad \sum_{j \in J(I)} p_{ij} x_{j,q_{j,t_2}} \geq \xi(i, I), \qquad \text{and} \qquad \forall j, \forall q : \quad x_{j,q} \in \{0, 1\}.$$

In Appendix B we shall argue that the value of above IP is within $O(1)$ factor of the optimum cost of the scheduling problem.

The above integer program hints towards how we can interpret COSSP geometrically as PR2C. Indeed, it is equivalent to PR2C problem. For every machine i in COSSP, we create a set $\mathcal{P}_i$ in PR2C. For every interval $I = [t_1, t_2]$ with $\xi(i, I) > 0$, we associate a point $p_I = (t_1, t_2)$ in 2-dimensional space. The demand d_{p_I} of a point $p_I \in \mathcal{P}_i$ is equal to $\xi(i, I)$, the excess of interval I on machine i. This completes the description of the point sets $\mathcal{P}_1, \mathcal{P}_2, \ldots \mathcal{P}_m$ in PR2C. Now we define the rectangle set $\mathcal{R}$. We shall create a rectangle for each job j and each q for which $x_{j,q}$ is defined. Notice that $x_{j,q}$ appears on the left side of Constraint (11) if $r_j \in [t_1, t_2]$ and $t_2 \in (t_{j,q-1}, t_{j,q}]$. Thus, we shall let the rectangle for the variable $x_{j,q}$ to be $(0, r_j] \times (t_{j,q-1}, t_{j,q}]$. This rectangle has cost 2^q; for the instance $\mathcal{P}_i$, it has a capacity p_{ij}. Notice that the rectangle for $x_{j,q}$ covers a point (t_1, t_2) if and only if $t_1 \leq r_j \leq t_{j,q-1} < t_2 \leq t_{j,q}$; in other words, the job j is released in the interval $[t_1, t_2]$, and its completion time is greater than t_2, which is exactly what we want. Thus, the IP (10 - 12) is equivalent to our PR2C problem. We now forget about the COSSP problem and focus exclusively on designing a good algorithm for the PR2C problem.

2.2 Algorithm For PR2C Problem

Our algorithm for PR2C is based on rounding a linear programming relaxation of the IP (1 - 3). As pointed out in [11], the linear programing relaxation of the IP (1 - 3) obtained by relaxing the variables $x_r \in [0, 1]$ has a large integrality gap even when there is only one set of points. We strengthen our LP by adding the so called KC inequalities, introduced first in the work of Carr *et al* ([14]). Towards that we need to define $c(i, S)$, which indicates the

total capacity of rectangles in $S \subseteq \mathcal{R}$ with respect to a point set $\mathcal{P}_i$: $c(i,S) = \sum_{r \in S} c(i,r)$. We are ready to write the LP.

$$\text{Minimize} \quad \sum_{r \in \mathcal{R}} w(r)x_r \qquad \text{s.t.} \tag{4}$$

$$\forall i, \forall p \in P_i, \forall S \subseteq \mathcal{R} : \sum_{r \in \mathcal{R} \setminus S : p \in r} \min\{c(i,r), \max\{0, d_p - c(i,S)\}\} \cdot x_r \geq d_p - c(i,S) \tag{5}$$

$$\forall r \in \mathcal{R} : \quad x_r \geq 0 \tag{6}$$

Let us focus on the KC inequalities Eq.(5) for a point set $\mathcal{P}_i$. Fix a point $p \in \mathcal{P}_i$ and a set of rectangles $S \in \mathcal{R}$. Recall that p has a demand of d_p. Suppose, in an integral solution all the rectangles in S are chosen. Then, they contribute at most $c(i,S)$ towards satisfying the demand of the point p. The remaining rectangles still need to account for $d_p - c(i,S)$. Notice also that in the KC constraints we truncate the capacity of a rectangle r to $\min\{c(i,r), \max\{0, d_p - c(i,S)\}\}$. This ensures that the LP solution does not cheat by picking a rectangle with a large capacity to a tiny fraction. Clearly, the truncation has no effect on the integral solution.

There are exponentially many KC constraints in our LP. However, using standard arguments, we can solve the LP in polynomial time to get a $(1+\epsilon)$-approximate solution, for any $\epsilon > 0$, which suffices for our purposes to obtain a logarithmic approximation to COSSP; see [11, 14] for more details. The rest of this section is devoted to rounding the LP solution.

Weighted Geometric Set Multi-Cover Problem. The main tool used in our rounding algorithm is the result by [12] for the *weighted geometric set multi-cover problem*. Similar to the standard set cover problem, in this problem, we are given a set U of points and a set M consisting of subsets of U; typically, the sets in M are defined by geometric objects. Further, each point $p \in U$ has a demand d_p, and a set $r \in M$ has a weight $w(r)$. The goal is to find the minimum weight set $S \subseteq M$ such that every point p is covered by at least d_p number of subsets in S. Note crucially that the sets in M do not have capacities. If the sets have capacities, then the problem becomes similar to PR2C. The interesting aspect of geometric set cover problems is that sets in M are defined by geometric objects, hence subsets in M have more structure than in the standard set cover problem. In particular, if the sets in M have low union complexity, then the problem admits a better than $O(\log |M|)$ approximation algorithm. Now we introduce the concept of union complexity of sets.

▶ **Definition 7.** Given a set S of n geometric objects, the union complexity of S is the number of edges in the arrangement of the boundary of S.

We will not get into the details of the definition, and we refer the readers interested in knowing more to [44] for an excellent introduction or to [50, 15, 12]. For our purposes, it suffices to know that the union complexity of 3-dimensional objects is the *total number of vertices, edges, and faces on the boundary*. It turns out that the geometric objects in our rounding algorithm are 4-dimensional objects. However, by appropriate projection, we reduce our problem of bounding the union complexity of 4-dimensional objects to that of 3-dimensional ones.

For geometric objects with low union complexity, building on the breakthrough works of Chan *et al* [15] and Varadarajan [50], Bansal and Pruhs [12] showed the following theorem.

▶ **Theorem 8.** *Suppose the union complexity of every $S \subseteq M$ of size k is at most $kh(k)$. Then, there is a polynomial time $O(\log h(n))$ approximation algorithm for the weighted geometric set multi-cover problem. Further, the approximation factor holds even against a feasible fractional solution.*

Rounding: Our rounding algorithm is based on reducing our problem to several instances of the geometric set cover problem, and then appealing to Theorem 8. Consider an optimal solution $\vec{x} = \{x_r\}_r$ to the LP (4 - 6). Let $\beta > 1/20$ be some constant. We first scale the solution by a factor of $1/\beta$, and let $\vec{x'} = \{x'_r\}_r$ be this scaled solution. Clearly, scaling increases the cost of LP solution at most by a factor of $1/\beta$. Our rounding consists of three steps.

In the first step, we pick all rectangles r for which $x'_r \geq 1$. Let $\mathcal{S}$ denote this set. Let $\mathcal{P}_i(\mathcal{S})$ denote the set of points that are covered by rectangles in $\mathcal{S}$. We modify the point sets $\mathcal{P}_i$ for $i = 1, 2, ..., m$, by removing the points that are already covered by $\mathcal{S}$. For all $i \in [m]$, let $\mathcal{P}'_i = \mathcal{P}_i \setminus S$. In the second step, for each $\mathcal{P}'_i$ we classify the points in the set into two types, *heavy* or *light*, based on the LP solution. For heavy points, we create an instance of the geometric set cover problem, and then appeal to Theorem 8. The main technical difficulty here is to bound the union complexity of the geometric objects in our instance. For the light points, we create another separate instance of the geometric set multi-cover problem, and then apply the Theorem 8. Finally, we obtain our solution by taking the union of the rectangles picked in all three steps.

Fix a set $\mathcal{P}'_i$ of uncovered points. For a point $p \in \mathcal{P'}_i$, let $\mathcal{S}(p)$ denote the set of rectangles in $\mathcal{S}$ that contain p. That is, $\mathcal{S}(p) = \mathcal{S} \cap \{r : r \in \mathcal{R}, p \in r\}$. For a point p, define the residual demand of p as $d_p - c(i, \mathcal{S}_p)$. From the definition of set $\mathcal{P}'_i$, for every point $p \in \mathcal{P}'_i$, we note that $d_p - c(i, \mathcal{S}_p) > 0$. We now apply the KC inequalities on the set $\mathcal{S}(p)$ for each point p. From Eq.(5) we have, for all $p \in \mathcal{P}'_i$:

$$\sum_{r \in \mathcal{R} \setminus \mathcal{S}(p): p \in r} (\min\{c(i, r), d_p - c(i, \mathcal{S}(p))\}) \cdot x_r, \geq d_p - c(i, \mathcal{S}(p))$$

which implies that our scaled solution $\vec{x'}$ satisfies for all $p \in \mathcal{P}'_i$:

$$\sum_{r \in \mathcal{R} \setminus \mathcal{S}(p): p \in r} (\min\{c(i, r), d_p - c(i, \mathcal{S}(p))\}) \cdot x'_r \geq \frac{d_p - c(i, \mathcal{S}(p))}{\beta}.$$

Note that for all r, $x'_r \in [0, 1]$; otherwise, we would have picked those rectangles in $\mathcal{S}$. Next we round the residual demands of points and the capacities of rectangles as follows: For a point $p \in \mathcal{P}'_i$, let $\widetilde{d}_p$ denote the residual demand of p rounded up to the nearest power of 2. On the other hand, we round down the capacities $c(i, r)$ of rectangles $r \in \mathcal{R}$ to the nearest power of 2; let $\widetilde{c}(i, r)$ denote this new rounded down capacities. Since $\vec{x'}$ is scaled by a factor of $1/\beta$ we still have that for all $p \in \mathcal{P}'_i$,

$$\textstyle\sum_{r \in \mathcal{R} \setminus \mathcal{S}(p): p \in r} \left(\min\{\widetilde{c}(i, r), \widetilde{d}_p\}\right) \cdot x'_r \geq \frac{\widetilde{d}_p}{4\beta} \geq 3\widetilde{d}_p. \tag{7}$$

To classify the points into heavy and light, we need the notion of class.

▶ **Definition 9.** Let $\widetilde{c}_{\min}(i) = \min_{r \in \mathcal{R}}\{\widetilde{c}(i, r)\}$. A rectangle $r \in \mathcal{R}$ is a class k rectangle with respect to a point set $\mathcal{P}_i$ if $\widetilde{c}(i, r) = 2^k \cdot \widetilde{c}_{\min}(i)$. We say that a point $p \in \mathcal{P'}_i$ is a class k point if $\widetilde{d}_p = 2^k \cdot \widetilde{c}_{\min}(i)$.

Recall that the capacities of rectangles for different point sets can be different. Therefore, the class of a rectangle depends on the point set $\mathcal{P'}_i$. Now we categorize points in $\mathcal{P'}_i$ into heavy and light as follows.

▶ **Definition 10.** A point $p \in \mathcal{P'}_i$ belonging to class k is heavy if its demand is satisfied by the rectangles of class at least k in the LP solution. Otherwise, we say that the point is light.

From the definition, for all heavy points $p \in \mathcal{P}'_i$ we have

$$\sum_{r \in \mathcal{R} \setminus \mathcal{S} : \widetilde{c}(i,r) \geq \widetilde{d}_p, p \in r} x'_r \geq 1, \tag{8}$$

and from Eq.(7) all light points $p \in \mathcal{P}'_i$ satisfy

$$\sum_{r \in \mathcal{R} \setminus \mathcal{S} : \widetilde{c}(i,r) < \widetilde{d}_p, p \in r} \widetilde{c}(i,r) x'_r \geq (1/4\beta - 1)\widetilde{d}_p = 2\widetilde{d}_p. \tag{9}$$

Let $\mathcal{P}^1_i \subseteq \mathcal{P}'_i$ denote the set of heavy points and let $\mathcal{P}^2_i \subseteq \mathcal{P}'_i$ denote the set of light points. Let $\mathcal{R}' = \mathcal{R} \setminus \mathcal{S}$. We create two separate instances, a heavy instance and a light instance of the PR2C problem, corresponding to the set of heavy points and the set of light points. The capacities of rectangles in these instances will be their rounded down values: that is, the capacity of a rectangle $r \in \mathcal{R}'$ for the point sets $\mathcal{P}^*_i$ will be $\widetilde{c}(i,r)$. Similarly, the demand of a point $p \in P^*_i$ will be its residual demand $d_p - c(*, \mathcal{S}(p))$. Notice that the LP solution $\vec{x'}$ restricted to $\mathcal{R}'$ is a feasible solution for both $\{\mathcal{P}^1_i\}_i$ and $\{\mathcal{P}^2_i\}_i$. We round the solution $\vec{x}'$ for these two instances of PR2C problem separately, and take their union. The algorithm for the light instance will be deferred in the Appendix C; for the instance we get an $O(\log m)$ approximation ratio. We now show how to solve the heavy instance.

2.3 Heavy Instance and Hyper-4cuboid Covering Problem

We round the heavy instance $(\{\mathcal{P}^1_i\}_i, \mathcal{R}')$ by reducing it to a weighted geometric set cover problem called hyper-4cuboid covering problem (HCCP). HCCP is a generalization of the R3U problem considered in [11]. In HCCP, we are given a set U of points $p = (p_1, p_2, p_3, p_4)$ in 4-dimensional space. We are also given a set M of geometric objects. Each object $v \in M$ is a set of m disjoint 4-dimensional cuboids. Formally, $v = \{r_1, r_2, \ldots r_m\}$, where each r_i is a 4-dimensional cuboid (4-cuboid) of the form $[2i, 2i+1] \times [0, x] \times [y_1, y_2] \times [0, z]$. We call v a hyper-4cuboid. Each hyper-4cubiod has a cost $w(v)$. The goal is to find the minimum cost set $S \subseteq M$ such that S covers all the points in U. We say that a hyper-4cuboid v covers a point p if $\exists r_i \in v, p \in r_i$.

Now we show that there is a reduction from the PR2C problem on the heavy instance to the HCCP problem. For a point $p \in \mathcal{P}^1_i$ with coordinates (x, y), we create a point $p' \in U$ with coordinates $(2i + 1/2, x, y, \widetilde{d}_p)$. Note that the last coordinate of p' is determined by $\widetilde{d}_p$, which is the residual demand of point p. And the first coordinate of p' is determined by the index of the set $\mathcal{P}^1_i$. For every rectangle $r \in \mathcal{R}'$, we create a hyper-4cuboid $v_r \in M$, which contains exactly one 4cuboid for each point set $\mathcal{P}^1_i$. For a given index $i \in [m]$ and a rectangle $r = [0, x] \times [y_1, y_2]$, there is a 4cuboid r_i of the form $[2i, 2i+1] \times [0, x] \times [y_1, y_2] \times [0, \widetilde{c}(i, r)]$. Note that the last coordinate is determined by the capacity of rectangle r towards the point set $\mathcal{P}^1_i$. The cost of hyper-4cuboid v_r is same as the cost of rectangle r.

▶ **Lemma 11.** *Suppose there is a feasible solution of cost α to the* PR2C *problem on the heavy instance, where for each $i \in [m]$ and for each $p \in \mathcal{P}^1_i$, the demand of the point p is completely satisfied by the rectangles of class at least the class of p. Then there is a feasible solution of cost at most α for the corresponding instance of* HCCP. *Similarly, a solution of cost α to the* HCCP *problem gives a solution of the same cost for the heavy instance of* PR2C.

We defer the proof of the lemma to Appendix D. Now, observe crucially that there is a fractional solution of cost at most the cost of LP solution to the heavy instance satisfying the requirements of Lemma 11. This is true because the LP solution $\vec{x}'$ satisfies the inequality Eq.(8). Therefore to apply Theorem 8, it remains to quantify the union complexity of hyper-4cuboids in our reduction.

▶ **Lemma 12.** *The union complexity of any q hyper-4 cuboids in M is at most $O(qm \log P)$, where $P = \frac{\max\{p_{ij}\}}{\min\{p_{ij}\}}$.*

Proof. Consider a set $S \subseteq M$ of q hyper-4 cuboids. For $i = 1, 2, \ldots m$, define $\mathcal{L}_i$ as the set of all 4-cuboids at level i; Formally, $\mathcal{L}_i \subseteq \{\bigcup_{i,v \in S} r_i(v)\}$, such that every 4-cuboid $r \in \mathcal{L}_i$ has the first dimension equal to $[2i, 2i + 1]$. In other words, $\mathcal{L}_i$ is the set of all 4-cuboids which have the same first dimension $[2i, 2i + 1]$. Now, observe from our construction that for any pair i, i', the objects in $\mathcal{L}_i$ and $\mathcal{L}_{i'}$ do not intersect. This is because 4-cuboids at different levels are separated by a distance of 1 in the first dimension. Therefore, the union complexity of the subset S is equal to m times the maximum of union complexity of 4-cuboids in $\{\mathcal{L}_i\}_i$. However, 4-cuboids in the sets $\mathcal{L}_i$ all have the same form; thus, it suffices to bound the union complexity for any $\mathcal{L}_i$.

Fix some i, and consider the 4-cuboids in the set $\mathcal{L}_i$. Notice carefully that all the 4-cuboids in $\mathcal{L}_i$ share the same first dimension $[2i, 2i + 1]$. This implies that we can ignore the first dimension as it does not add any edges to the arrangement of objects in $\mathcal{L}_i$. Now consider the projection of 4-cuboids to the remaining 3 dimensions; These are cuboids of the form $[0, x] \times [y_1, y_2] \times [0, z]$. For these type of cuboids, [11] proves that the union complexity is at most $O(q\theta)$, where θ is the number of distinct values taken by z. In our reduction, the values of z correspond to the capacities of rectangles. Recall that the capacities of rectangles in PR2C are defined based on the processing lengths of operations. As we round down the capacities to the nearest power of 2, the number of distinct values taken by z is at most $\log P$, where $P = \frac{\max\{p_{ij}\}}{\min\{p_{ij}\}}$. Putting everything together, we complete the proof. ◀

From Lemma 11, 12 and Theorem 8, we get the following.

▶ **Lemma 13.** *There is a solution of cost $O(\log(m \log P))$ times the cost of LP solution for the* HCCP *problem, and hence for the* PR2C *on the heavy instance.*

Proof of Theorem 1. Our final solution for PR2C problem is obtained by taking the union of all the rectangles picked in our solutions for the heavy and the light instances, and the set $\mathcal{S}$. Recall that in $\mathcal{S}$, we pick all the rectangles for which $x_r > 1/\beta$, for $\beta = 12$. The cost of our solution is at most $(O(\log(m \log P)) + O(\log m) + 1/\beta))$ times the cost of LP (4 - 6), which implies an $O(\log(m \log P))$ approximation algorithm. This completes the proof. ◀

References

1. Apache hadoop.
2. Kunal Agrawal, Jing Li, Kefu Lu, and Benjamin Moseley. Scheduling parallel DAG jobs online to minimize average flow time. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 176–189, 2016. doi:10.1137/1.9781611974331.ch14.
3. Reza Ahmadi, Uttarayan Bagchi, and Thomas A Roemer. Coordinated scheduling of customer orders for quick response. *Naval Research Logistics (NRL)*, 52(6):493–512, 2005.
4. Saba Ahmadi, Samir Khuller, Manish Purohit, and Sheng Yang. On scheduling coflows. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 13–24. Springer, 2017.
5. S. Anand, Naveen Garg, and Amit Kumar. Resource augmentation for weighted flow-time explained by dual fitting. In *SODA*, pages 1228–1241, 2012. URL: http://portal.acm.org/citation.cfm?id=2095213{&}CFID=63838676{&}CFTOKEN=79617016.
6. Nikhil Bansal, Ho-Leung Chan, Rohit Khandekar, Kirk Pruhs, Cliff Stein, and Baruch Schieber. Non-preemptive min-sum scheduling with resource augmentation. In *Foundations*

of Computer Science, 2007. FOCS'07. 48th Annual IEEE Symposium on, pages 614–624. IEEE, 2007.

7 Nikhil Bansal, Moses Charikar, Ravishankar Krishnaswamy, and Shi Li. Better algorithms and hardness for broadcast scheduling via a discrepancy approach. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 55–71. Society for Industrial and Applied Mathematics, 2014.

8 Nikhil Bansal and Subhash Khot. Inapproximability of hypergraph vertex cover and applications to scheduling problems. *Automata, Languages and Programming*, pages 250–261, 2010.

9 Nikhil Bansal, Ravishankar Krishnaswamy, and Viswanath Nagarajan. Better scalable algorithms for broadcast scheduling. In *ICALP (1)*, pages 324–335, 2010. doi:10.1007/978-3-642-14165-2_28.

10 Nikhil Bansal and Janardhan Kulkarni. Minimizing flow-time on unrelated machines. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 851–860, 2015. doi:10.1145/2746539.2746601.

11 Nikhil Bansal and Kirk Pruhs. The geometry of scheduling. In *IEEE Symposium on the Foundations of Computer Science*, pages 407–414, 2010.

12 Nikhil Bansal and Kirk Pruhs. Weighted geometric set multi-cover via quasi-uniform sampling. In *European Symposium on Algorithms*, pages 145–156. Springer, 2012.

13 Jatin Batra, Naveen Garg, and Amit Kumar. Constant factor approximation algorithm for weighted flow time on a single machine in pseudo-polynomial time. *CoRR*, abs/1802.07439, 2018.

14 Robert D Carr, Lisa Fleischer, Vitus J Leung, and Cynthia A Phillips. Strengthening integrality gaps for capacitated network design and covering problems. In *SODA*, pages 106–115, 2000.

15 Timothy M Chan, Elyot Grant, Jochen Könemann, and Malcolm Sharpe. Weighted capacitated, priority, and geometric set cover via improved quasi-uniform sampling. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 1576–1585. Society for Industrial and Applied Mathematics, 2012.

16 Zhi-Long Chen and Nicholas G Hall. Supply chain scheduling: Conflict and cooperation in assembly systems. *Operations Research*, 55(6):1072–1089, 2007.

17 Mosharaf Chowdhury and Ion Stoica. Coflow: A networking abstraction for cluster applications. In *Proceedings of the 11th ACM Workshop on Hot Topics in Networks*, pages 31–36. ACM, 2012.

18 Mosharaf Chowdhury and Ion Stoica. Efficient coflow scheduling without prior knowledge. In *ACM SIGCOMM Computer Communication Review*, volume 45, pages 393–406. ACM, 2015.

19 Mosharaf Chowdhury, Yuan Zhong, and Ion Stoica. Efficient coflow scheduling with varys. In *ACM SIGCOMM Computer Communication Review*, volume 44, pages 443–454. ACM, 2014.

20 Nikhil Devanur and Janardhan Kulkarni. A unified rounding algorithm for unrelated machines scheduling. In *Preprint, https://users.cs.duke.edu/ kulkarni/papers/tardines.pdf*, 2017.

21 Devdatta Gangal and Abhiram Ranade. Precedence constrained scheduling in $(2-\frac{7}{3}p+1)\cdot$ optimal. *Journal of Computer and System Sciences*, 74(7):1139–1146, 2008.

22 Naveen Garg and Amit Kumar. Minimizing average flow-time : Upper and lower bounds. In *FOCS*, pages 603–613, 2007. doi:10.1109/FOCS.2007.42.

23 Naveen Garg, Amit Kumar, and Vinayaka Pandit. Order scheduling models: hardness and algorithms. *FSTTCS 2007: Foundations of Software Technology and Theoretical Computer Science*, pages 96–107, 2007.

24 Ronald L Graham. Bounds for certain multiprocessing anomalies. *Bell Labs Technical Journal*, 45(9):1563–1581, 1966.

25 Robert Grandl, Mosharaf Chowdhury, Aditya Akella, and Ganesh Ananthanarayanan. Altruistic scheduling in multi-resource clusters. In *OSDI*, pages 65–80, 2016.

26 Robert Grandl, Srikanth Kandula, Sriram Rao, Aditya Akella, and Janardhan Kulkarni. GRAPHENE: packing and dependency-aware scheduling for data-parallel clusters. In *12th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2016, Savannah, GA, USA, November 2-4, 2016.*, pages 81–97, 2016. URL: `https://www.usenix.org/conference/osdi16/technical-sessions/presentation/grandl_graphene`.

27 Sungjin Im, Janardhan Kulkarni, and Kamesh Munagala. Competitive algorithms from competitive equilibria: non-clairvoyant scheduling under polyhedral constraints. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 313–322, 2014. `doi:10.1145/2591796.2591814`.

28 Sungjin Im, Janardhan Kulkarni, and Kamesh Munagala. Competitive flow time algorithms for polyhedral scheduling. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 506–524, 2015. `doi:10.1109/FOCS.2015.38`.

29 Sungjin Im, Janardhan Kulkarni, Kamesh Munagala, and Kirk Pruhs. Selfishmigrate: A scalable algorithm for non-clairvoyantly scheduling heterogeneous processors. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 531–540, 2014. `doi:10.1109/FOCS.2014.63`.

30 Sungjin Im, Shi Li, Benjamin Moseley, and Eric Torng. A dynamic programming framework for non-preemptive scheduling problems on multiple machines. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1070–1086. Society for Industrial and Applied Mathematics, 2015.

31 Sungjin Im and Benjamin Moseley. Online scalable algorithm for minimizing ;k-norms of weighted flow time on unrelated machines. In *SODA*, pages 95–108, 2011. URL: `http://www.siam.org/proceedings/soda/2011/SODA11_008_ims.pdf`.

32 Sungjin Im and Benjamin Moseley. An online scalable algorithm for average flow time in broadcast scheduling. *ACM Transactions on Algorithms*, 8(4):39, 2012. `doi:10.1145/2344422.2344429`.

33 Sungjin Im and Benjamin Moseley. General profit scheduling and the power of migration on heterogeneous machines. In *Proceedings of the 28th ACM Symposium on Parallelism in Algorithms and Architectures, SPAA 2016, Asilomar State Beach/Pacific Grove, CA, USA, July 11-13, 2016*, pages 165–173, 2016.

34 Bala Kalyanasundaram and Kirk Pruhs. Speed is as powerful as clairvoyance. *Journal of the ACM*, 47(4):617–643, 2000.

35 Bala Kalyanasundaram and Kirk R Pruhs. Eliminating migration in multi-processor scheduling. *Journal of Algorithms*, 38(1):2–24, 2001.

36 Samir Khuller and Manish Purohit. Brief announcement: Improved approximation algorithms for scheduling co-flows. In *Proceedings of the 28th ACM Symposium on Parallelism in Algorithms and Architectures*, pages 239–240. ACM, 2016.

37 Shui Lam and Ravi Sethi. Worst case analysis of two scheduling algorithms. *SIAM Journal on Computing*, 6(3):518–536, 1977.

38 J. K. Lenstra and A. H. G. Rinnooy Kan. Complexity of scheduling under precedence constraints. *Oper. Res.*, 26(1):22–35, 1978. `doi:10.1287/opre.26.1.22`.

39 Joseph Y-T Leung, Haibing Li, and Michael Pinedo. Scheduling orders for multiple product types to minimize total weighted completion time. *Discrete Applied Mathematics*, 155(8):945–970, 2007.

40 Elaine Levey and Thomas Rothvoss. A (1+ epsilon)-approximation for makespan scheduling with precedence constraints using lp hierarchies. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing*, pages 168–177. ACM, 2016.

41 Shi Li. Scheduling to minimize total weighted completion time via time-indexed linear programming relaxations. In *Proceedings of the 2017 IEEE 58rd Annual Symposium on Foundations of Computer Science*, FOCS '17, 2017.

42 Pasin Manurangsi. Almost-polynomial ratio eth-hardness of approximating densest k-subgraph. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 954–961, 2017. doi:10.1145/3055399.3055412.

43 Monaldo Mastrolilli, Maurice Queyranne, Andreas S Schulz, Ola Svensson, and Nelson A Uhan. Minimizing the sum of weighted completion times in a concurrent open shop. *Operations Research Letters*, 38(5):390–395, 2010.

44 Jiří Matoušek. *Lectures on discrete geometry*, volume 212. Springer Science & Business Media, 2002.

45 Alix Munier, Maurice Queyranne, and Andreas S. Schulz. *Approximation Bounds for a General Class of Precedence Constrained Parallel Machine Scheduling Problems*, pages 367–382. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998. doi:10.1007/3-540-69346-7_28.

46 Zhen Qiu, Cliff Stein, and Yuan Zhong. Minimizing the total weighted completion time of coflows in datacenter networks. In *Proceedings of the 27th ACM symposium on Parallelism in Algorithms and Architectures*, pages 294–303. ACM, 2015.

47 Maurice Queyranne and Andreas S. Schulz. Approximation bounds for a general class of precedence constrained parallel machine scheduling problems. *SIAM J. Comput.*, 35(5):1241–1253, 2006. doi:10.1137/S0097539799358094.

48 Mehrnoosh Shafiee and Javad Ghaderi. An improved bound for minimizing the total weighted completion time of coflows in datacenters. *arXiv preprint arXiv:1704.08357*, 2017.

49 Ola Svensson. Conditional hardness of precedence constrained scheduling on identical machines. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 745–754. ACM, 2010.

50 Kasturi Varadarajan. Epsilon nets and union complexity. In *Proceedings of the twenty-fifth annual symposium on Computational geometry*, pages 11–16. ACM, 2009.

51 Edouard Wagneur and Chelliah Sriskandarajah. Openshops with jobs overlap. *European Journal of Operational Research*, 71(3):366–378, 1993.

52 Guoqing Wang and TC Edwin Cheng. Customer order scheduling to minimize total weighted completion time. *Omega*, 35(5):623–626, 2007.

53 Jaehwan Yang. *Scheduling with batch objectives*. PhD thesis, Ohio State University, 1998.

A Applications of COSSP and PCSP in Data Center Scheduling

Besides being fundamental optimization problems, COSSP and PCSP models are very closely related to the scheduling problems that arise in the context of data centers. We briefly describe the relevance of COSSP and PCSP to scheduling in data centers.

Coflow Scheduling. The COSSP problem is a special case of the *coflow scheduling abstraction* introduced in a very influential work of Choudary and Stoica [17, 19], in the context of scheduling data flows. They defined coflow as a collection of parallel flows with a

common performance goal. Their main motivation to introduce coflow was that in big-data systems, MapReduce systems for example, communication is structured and takes place across machines in successive computational stages. In most cases, communication stage of jobs cannot finish until all its flows have completed. For example, in MapReduce jobs, a reduce task cannot begin until *all* the map tasks finish. Although coflow abstraction was introduced to model scheduling flows in big data networks, it can also be applied to job scheduling in clusters; see [25] for example.

Therefore, we describe a slightly more general version of the coflow abstraction. Here, we are given a set of m machines (or m resources). Each job j consists of a set of operations $\{O_{ij}\}$ for $i = 1, 2....$ Associated with each operation O_{ij} is a demand vector $D_{ij} = (d_{ij}(1), d_{ij}(2), ..., d_{ij}(m))$, where each $d_{ij}(k) \in \{0, 1\}$. The demand vector indicates the subset of machines or resources the operation requires. An operation can be executed only when all the machines in the demand vector are allocated to it. Moreover, for each operation we are also given a processing length p_{ij}. The goal is to schedule all operations such that at any time instant the capacity constraints on machines are not violated: that is, each machine is allocated to exactly one operation. A job completes only when all its operations have finished.

The coflow problem studied by Choudary and Stoica [17, 19] corresponds to the case where the demand vectors of all jobs have exactly two 1s. That is, every operation needs two machines to execute. The machines typically correspond to input and output ports in a communication link. On the other hand, if the demand vector D_{ij} of each operation O_{ij} consists of exactly one non-zero entry, then the coflow scheduling is equivalent to COSSP.

In the past few years, coflow scheduling has attracted a lot of research both in theory and systems communities. In practice, several heuristics are known to perform well [17, 19, 18, 25] for the problem. The theoretical study of coflow scheduling was initiated by Qiu, Stein, and Zhong [46]. By exploiting its connections to COSSP, they designed a constant factor approximation algorithm to the objective of minimizing the total weighted completion times of jobs. Building on this work, better approximation algorithms were designed in [36, 4, 48]. Unfortunately, the techniques developed in these works do not seem to extend to the flow-time related objectives.

DAG Scheduling. Another problem that has attracted a lot of research in practice is the DAG scheduling problem. In this problem, we are given a set of machines (or clusters), and a set of jobs. Each job has a weight that captures the priority of a job. Each job j is represented by a directed acyclic graph (DAG). Each node of a DAG represents a task – a single unit of computation– that needs to executed on a *single machine*. Each task has a release time and a processing length. An edge $j \to j'$ in the DAG indicates that the task j' depends on the task j, and j' can not begin its execution unless j finishes. The goal is to schedule jobs/DAGs on the machines so as to minimize the total weighted flow-time of jobs. Interestingly, this is one of the models of job scheduling that has been adopted by Yarn, the resource manager of Hadoop [1]. (Hadoop is a popular implementation of MapReduce framework.) Because of this, DAG scheduling has been a very active area of research in practice; see [26, 25] and references there-in for more details.

It is not hard to see that DAG scheduling problem is a special case of the precedence constrained scheduling problem. The union of the individual DAGs of jobs can be considered as one DAG, with appropriately defined release times and weights for each node. Furthermore, if jobs have no weight, and the release times of all tasks in the same DAG are equal, then the DAG scheduling model described above is same as the DAG parallelizability model

studied by Agrawal *et al* [2]. In fact, they design $(1+\epsilon)$-speed $O(1/\epsilon)$-competitive *online algorithm* for the problem. On the other hand, our approximation algorithm for PCSP gives an approximation algorithm for the DAG scheduling problem even with weights and arbitrary release times for individual tasks.

B Integer Program for COSSP

Our IP for COSSP is as follows:

$$\text{Minimize} \quad \sum_j \sum_q \lfloor 2^q \rfloor \, x_{j,q} \qquad \text{s.t.} \tag{10}$$

$$\forall i, \forall I = [t_1, t_2] : \ \sum_{j \in J(I)} p_{ij} x_{j,q_{j,t_2}} \geq \xi(i,I), \tag{11}$$

$$\forall j, \forall q : \ x_{j,q} \in \{0,1\}. \tag{12}$$

We shall argue that the value of above IP is within a factor of $O(1)$ times the optimum cost of the scheduling problem. First, given a feasible schedule S for COSSP with completion time vector $(C_j)_j$, we construct a solution to the above integer program with a small cost. For each j and integer q such that $2^{q-1} \leq g_j(C_j)$, we let $x_{j,q} = 1$; set all other x variables to 0. Clearly, the cost of the solution to the IP is at most $O(1)$ times the cost of S. Moreover, Constraint (11) is satisfied: we have that $\sum_{j \in J(I): C_j > t_2} p_{i,j} \geq \xi(i,I)$. If $C_j > t_2$ for some $j \in J(I)$, then $x_{j,q_{j,t_2}} = 1$, as $2^{q_{j,t_2}-1} \leq g_j(t_2) \leq g_j(C_j)$.

On the other hand, if we are given an optimum solution to the above IP, we can convert it into a schedule for COSSP of cost at most the cost of the IP. For every j, let the completion time C_j of the job to be $t_{j,q}$, where q is the largest number such that $x_{j,q} = 1$; such a q must exist in order to satisfy the Constraint (11) for the interval $t_1 = t_2 = r_j$. Then the cost of our schedule is at most the cost of IP. On the other hand, Constraint (11) says that $\sum_{j \in J(I)} p_{ij} x_{j,q_{j,t_2}} \geq \xi(i,I)$, implying $\sum_{j \in J(I), C_j > t_2} p_{ij} \geq \xi(i,I)$, as $x_{j,q_{j,t_2}} = 1$ implies $C_j \geq t_2$.

▶ Remark. We will not discuss the representation of arbitrary delay functions $g_j(t)$ and how to compute $t_{j,q}$ here. We refer the readers to [11] for more details. Running time of all our algorithms will be polynomial in $n, m, \log L$ and $\max_j \{\log g_j(L)\}$.

C Covering Light Points and Geometric Multi Cover by Hyper-cuboids

Here, we design an $O(\log m)$ approximation algorithm to PR2C problem restricted to the light instance: $(\{\mathcal{P}_i^2\}_i, \mathcal{R}')$. There is one main technical difference between our algorithm for the light case from the algorithm of Bansal and Pruhs [11]. Bansal and Pruhs divide the light instance into $\log P$ different levels, where in each level they solve 2-dimensional geometric multi-cover problem, and show a $O(1)$ approximation to each level. This in turn leads to a constant factor approximation algorithm to the light instance. However, in our problem we cannot solve the instances separately. Therefore, using a simple trick, we map our problem into a single instance of 3-dimensional problem, and show a $(\log m)$ approximation to it. This is precisely the reason we loose the factor $O(\log m)$ for the light case, as the union complexity of our objects becomes m times the union complexity of the objects in [11].

Recall that for every point $p \in \mathcal{P}_i^2$, the LP solution satisfies the inequality Eq.(9). Again, our idea is to reduce the instance to a geometric uncapacitated set multi-cover problem, and then appeal to Theorem 8. The geometric objects produced by our reduction are *sets of cuboids*. Hence we abbreviate this problem as GMCC.

In GMCC, we are given a set U of points in 3-dimensional space, and a set M of geometric objects. Each geometric object $v \in M$ is a collection of m disjoint cuboids, which we call as hyper-cuboids. Each point $p \in U$ has a demand e_p, and each hyper-cuboid

$v \in M$ has a cost b_v. The individual cuboids constituting a hyper-cuboid v have the form $[x, x+1] \times [0, y] \times [z_1, z_2]$. The objective is to find the minimum cost subset $S \subseteq M$ of hyper-cuboids such that for every point $p \in U$, there are at least e_p number of hyper-cuboids in S that contain the point p.

Now we give the reduction $\mathcal{H}$ from an instance of PR2C to an instance of GMCC. Fix some $i \in [m]$. Let $T > 0$ be some large constant, and recall $P = \frac{\max\{p_{ij}\}}{\min\{p_{ij}\}}$. For every light point $p \in \mathcal{P}_i^2$ with coordinates (x_p, y_p), we create $\log P$ different points, each of them shifted in the second dimension by a distance of T. The $\log P$ different points corresponding to a point $p \in \mathcal{P}_i^2$ have coordinates $(2i + 1/2, kT + x_p, y_p)$, for $k = 1, 2, ... \log P$. To complete the description of point set U, we need to define the demands of each point, which we will do after describing the cuboids in M.

For every rectangle $r \in \mathcal{R}'$, with dimensions $[0, r_j] \times [t_1, t_2]$, we create one hyper-cuboid $v_r \in M$. The cost of hyper-cuboid v_r is same the as the cost of rectangle r. The hyper-cuboid v_r contains m different cuboids, one for each point set $\mathcal{P}_1^2, \mathcal{P}_2^2, \dots \mathcal{P}_m^2$. Fix an index $i \in [m]$. The cuboid corresponding to the set $\mathcal{P}_i^2$ has dimensions $[2i, 2i+1] \times [0, kT + r_j] \times [t_1, t_2]$, where k denotes the class of rectangle r with respect to $\mathcal{P}_i^2$. Note that there is one-to-one correspondence between rectangles $r \in \mathcal{R}'$ to hyper-cuboids $v_r \in M$. Let $\mathcal{H}^{-1}(v_r) = r$; that is, the rectangle $r \in \mathcal{R}'$ that corresponds to the hyper-cuboid $v_r \in M$. We say that v is picked to an extent of x'_r in the LP solution $\vec{x'}$ to mean the extent to which the rectangle r is picked in the LP solution.

Fix a point $p \in U$. We say that p is contained in the hyper-cuboid $v \in M$, if it is contained in any of the m cuboids that constitute v. Now we are ready to define the demand e_p of a point p as $e_p = \left\lfloor \sum_{v:p \in v} x_{\mathcal{H}^{-1}(v)} \right\rfloor$.

To understand the above definition, let us consider a point $p \in U$ with coordinates $(2i + 1/2, kT + x, y)$. The cuboids that can contain p should have form $[2i, 2i+1] \times [0, kT + r_j] \times [t_1, t_2]$, where $kT + x < r_j$ and $y \in [t_1, t_2]$. These cuboids correspond to the class k rectangles in $\mathcal{R}$. Therefore, demand of a point $p \in U$ with coordinates $(2i + 1/2, kT + x, y)$ is exactly equal to the number of class k rectangles that cover the point $p \in \mathcal{P}_i^2$ in the LP solution.

This immediately tells us that the LP solution $\vec{x'}$ is a feasible fractional solution to the instance of GMCC problem produced by our reduction $\mathcal{H}$. Now, we show the opposite direction.

▶ **Lemma 14.** *If there is an integral solution to the instance of* GMCC *produced by our reduction $\mathcal{H}$ of cost at most α times the LP cost, then there is an integral solution of the same cost to the instance of* PR2C *on the light instance.*

Proof. Suppose $S \subset M$ is a solution to GMCC problem. We show that there is a corresponding solution to PR2C problem of exactly the same cost. Our solution is simple. For every hyper-cuboid $v \in S$, we pick the corresponding rectangle r in our solution S' to PR2C. From our reduction $\mathcal{H}$, the costs of hyper-cuboids is same as the costs of rectangles, which implies that solution costs of the two problems are the same. It remains to show that S' is a feasible solution for the light instance.

Fix a point set $\mathcal{P}_i$, and consider any point $p \in \mathcal{P}_i$. Let $p = (x, y)$. In our reduction, there are $\log P$ different points $p'_1, p'_2, ..., p'_{\log P}$, where the point p'_k has coordinates $(2i + 1/2, kT + x, y)$. The demand of $d_{p'_k}$ is exactly equal to the total number of class k rectangles that cover the point p in the LP solution. Recall that in our reduction $\mathcal{H}$, each rectangle $r \in \mathcal{R}$ produces exactly one cubiod at the level i (that is with first dimension $[2i, 2i+1]$) in the corresponding hyper-cuboid $v_r \in S'$. By abusing notation, let us call denote this cuboid by $h(r, i)$.

Let class of point p be k^*. The total capacity of rectangles that cover point $p \in \mathcal{P}_i$ in our solutin S' is $\sum_{k \le k^*} \sum_{r \in S' : r \in \text{cls}(k)} 2^k \cdot \widetilde{c}_{\min}(i)$.

From the one-to-one correspondence between the rectangles and the cuboids, the term $\sum_{k \le k^*} \sum_{r \in S' : r \in \text{cls}(k)} 2^k \cdot \widetilde{c}_{\min}(i)$ is exactly equal to the number of cuboids that cover the point p'_k, which is at least the demand $d_{p'_k}$. Now, in our reduction, the the demand of the point p'_k (which has coordinates $(2i + 1/2, kT + x, y)$) is exactly equal to the rounded down value of the total number of class k rectangles that cover the point p in the LP solution. Therefore,

$$\sum_{k \le k^*} \sum_{r \in S' : r \in \text{cls}(k)} 2^k \cdot \widetilde{c}_{\min}(i) \ge \sum_{k \le k^*} 2^k \cdot \widetilde{c}_{\min}(i) \cdot \left\lfloor \sum_{r \in S' : r \in \text{cls}(k)} x'_r \right\rfloor .$$

Simplifying the righthand side of the above equation,

$$\sum_{k \le k^*} 2^k \cdot \widetilde{c}_{\min}(i) \cdot \left\lfloor \sum_{r \in S' : r \in \text{cls}(k)} x'_r \right\rfloor \ge \sum_{k \le k^*} 2^k \cdot \widetilde{c}_{\min}(i) \cdot \left(\left(\sum_{r \in S' : r \in \text{cls}(k)} x'_r \right) - 1 \right)$$

$$\ge \sum_{k \le k^*} 2^k \cdot \widetilde{c}_{\min}(i) \cdot \left(\sum_{r \in S' : r \in \text{cls}(k)} x'_r \right) - 2^{k^*} \widetilde{c}_{\min}(i) \ge 2d'(p) - d'(p) = d'(p),$$

where the last inequality follows from the Eq.(9), and the fact that p is a light point. Therefore, S' is a feasible solution to GMCC. ◀

Thus, to complete our algorithm for the light instance, it remains to design an $O(\log m)$ approximation algorithm to GMCC. For that we once again plan to rely on the Theorem 8, which requires to us to bound the union complexity of objects in M.

▶ **Lemma 15.** *The union complexity of any q hyper-cuboids in M is at most $O(mq)$.*

Proof. Consider any $S \subseteq M$ of q hyper-cuboids. From our definition, each hyper-cuboid v is a set of m cuboids. For $i = 1, 2, ..., m$, define $\mathcal{L}_i$ as the set of all cuboids with first dimension $[2i, 2i + 1]$. Clearly, for any two indices i, i', the cuboids in $\mathcal{L}_i$ and $\mathcal{L}_{i'}$ do not intersect, as they are separated by a distance of 1 in the first dimension. Moreover, the cuboids in different $\mathcal{L}_i$ have same form except that they are shifted in the first dimension. Thus, the union complexity of q hyper-cuboids in S is at most m times the union complexity of q cuboids in $\mathcal{L}_i$, for any $i \in [m]$.

Fix some i and consider the set of q cuboids in $\mathcal{L}_i$. Since all of the q cuboids have exactly the same side in the first dimension, $[2i, 2i + 1]$, the union complexity of q cuboids in $\mathcal{L}_i$ is equal to the union complexity of the projection of the cuboids to the last two dimensions. This projection of the cubiods produces a set of axis parallel rectangles. Furthermore, these q axis parallel rectangles partition into $\log P$ sets, $Z_1, Z_2, \ldots Z_{\log P}$, such that no two rectangles from different sets intersect. This is true as these rectangles have coordinates $[0, kT]$ in the second dimension. In each set Z_k, the rectangles are abutting the Y-axis (or the axis of second dimension). Bansal and Pruhs [11] showed that the union of complexity such axis parallel rectangles abutting Y-axis is at most $O(|Z_k|)$. Therefore, the union complexity all the q axis parallel rectangles is at most $\sum_k |Z_k| = O(q)$. Thus we conclude that the union complexity q hyper-cuboids is equal to mq, which completes the proof. ◀

Thus, from Theorem 8, we get a $O(\log m)$ approximation algorithm for the light instance of PR2C.

D Proof of Lemma 11

Proof of Lemma 11. Suppose $S \subset \mathcal{R}'$ is a feasible solution of cost α satisfying the condition stated in the lemma. Then, we claim that the set of hyper-4 cuboids S' corresponding to the rectangles $r \in S$ is a feasible solution to HCCP. Consider a point $p' \in U$ in the instance of HCCP produced by our reduction. Suppose p' has coordinates $(2i+1/2, x, y, \widetilde{d}_p)$. Then, there must be a point $p \in \mathcal{P}_i^1$ with coordinates (x, y) and the demand $\widetilde{d}_p$. Suppose $r \in S$ covers this point p, and has dimensions $r = [0, x] \times [y_1, y_2]$. Then, we claim that $v_r \in S'$ covers the point p'. This is true, since v_r contains a 4-cuboid with coordinates $h = [2i, 2i+1] \times [0, x] \times [y_1, y_2] \times [0, \widetilde{c}(i, r)]$. It is easy to verify that $p' \in h$ as $\widetilde{c}(i, r) > \widetilde{d}_p$, which follows from the condition of the lemma.

The opposite direction also follows from the one-to-one correspondence between the rectangles in $\mathcal{R}$ and the hyper-4cuboids. Suppose S' is a feasible solution to the instance of HCCP problem. Then, it is easy to verify that picking the rectangles $r \in \mathcal{R}$ corresponding to the hyber-4cuboids $v_r \in S$ defines a feasible a solution to $(\{\mathcal{P}_i^1\}_i, \mathcal{R}')$. ◀

Sublinear-Time Quadratic Minimization via Spectral Decomposition of Matrices

Amit Levi[1]
University of Waterloo, Canada
amit.levi@uwaterloo.ca
https://orcid.org/0000-0002-8530-5182

Yuichi Yoshida[2]
National Institute of Informatics, Tokyo, Japan
yyoshida@nii.ac.jp
https://orcid.org/0000-0001-8919-8479

Abstract

We design a sublinear-time approximation algorithm for quadratic function minimization problems with a better error bound than the previous algorithm by Hayashi and Yoshida (NIPS'16). Our approximation algorithm can be modified to handle the case where the minimization is done over a sphere. The analysis of our algorithms is obtained by combining results from graph limit theory, along with a novel spectral decomposition of matrices. Specifically, we prove that a matrix A can be decomposed into a structured part and a pseudorandom part, where the structured part is a block matrix with a polylogarithmic number of blocks, such that in each block all the entries are the same, and the pseudorandom part has a small spectral norm, achieving better error bound than the existing decomposition theorem of Frieze and Kannan (FOCS'96). As an additional application of the decomposition theorem, we give a sublinear-time approximation algorithm for computing the top singular values of a matrix.

2012 ACM Subject Classification Theory of computation → Sketching and sampling, Theory of computation → Probabilistic computation

Keywords and phrases Qudratic function minimization, Approximation Algorithms, Matrix spectral decomposition, Graph limits

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.17

1 Introduction

Quadratic function minimization/maximization is a versatile tool used in machine learning, statistics, and data mining and can represent many fundamental problems such as linear regression, k-means clustering, principal component analysis (PCA), support vector machines, kernel machines and more (see [17]). In general, quadratic function minimization/maximization is NP-Hard. When the problem is convex (for minimization) or concave (for maximization), we can solve it by solving a system of linear equations, which requires $O(n^3)$ time, where n is the number of variables. There are faster approximation methods based on stochastic gradient descent [3], and the multiplicative update algorithm [5]. However, these methods still require $\Omega(n)$ time, which is prohibitive when we need to handle a huge number of variables.

[1] Research supported by NSERC Discovery grant and the David R. Cheriton Graduate Scholarship. Part of this work was done while the author was visiting NII Tokyo.
[2] Research supported by JSPS KAKENHI Grant Number JP17H04676 and JST ERATO Grant Number JPMJER1201.

Quadratic function minimization over a sphere is also an important problem. This minimization problem is often called the *trust region subproblem* since it must be solved in each step of a *trust region algorithm*. Trust region algorithms are among the most important tools in solving nonlinear programming problems, as they are robust and can be applied to ill-conditioned problems. In addition, trust region subproblems are useful in many other problems such as constrained eigenvalue problems [9], least-square problems [24], combinatorial optimization problems [4] and many more. While the problem is non-convex, it has been shown that the problem exhibit strong duality properties and is known to be solved in polynomial time (see [2, 23]). In particular, it was shown to be equivalent to some semidefinite programming optimization problems that can be solved in polynomial time ([19, 1]). As in the non-constrained case, there are approximation algorithms based on gradient descent [18] and on reducing the problem to a sequence of eigenvalues computations [12]. However, as in the unconstrained case, these methods require running time which is linear in the number of the non-zero elements of the matrix (which might be linear in n).

1.1 Our Contributions

In this work, we provide sublinear-time approximation algorithms for minimizing quadratic functions, assuming random access to the entries of the input matrix and the vector.

First, we consider unconstraind minimization. Specifically, for a matrix $A \in \mathbb{R}^{n\times n}$ and vectors $\boldsymbol{d}, \boldsymbol{b} \in \mathbb{R}^n$, we consider the following quadratic function minimization problem:

$$\min_{\boldsymbol{v}\in\mathbb{R}^n} \psi_{n,A,\boldsymbol{d},\boldsymbol{b}}(\boldsymbol{v}), \text{ where } \psi_{n,A,\boldsymbol{d},\boldsymbol{b}}(\boldsymbol{v}) = \langle \boldsymbol{v}, A\boldsymbol{v}\rangle + n\langle \boldsymbol{v}, \mathrm{diag}(\boldsymbol{d})\boldsymbol{v}\rangle + n\langle \boldsymbol{b}, \boldsymbol{v}\rangle \ . \tag{1}$$

Here $\mathrm{diag}(\boldsymbol{d}) \in \mathbb{R}^{n\times n}$ is a matrix whose diagonal entries are specified by $\boldsymbol{d}$ and $\langle\cdot,\cdot\rangle$ denotes the standard inner product.

▶ **Theorem 1.** *Fix $\epsilon > 0$ and let $\boldsymbol{v}^*$ and z^* be an optimal solution and the optimal value, respectively, of Problem (1). Let S be a random set such that each index $i \in \{1, 2, \ldots, n\}$ is taken to S independently w.p k/n with*

$$k = \max\left\{ O\left(\frac{\log^2 n}{\epsilon^2}\right), \left(\frac{1}{\epsilon}\right)^{O(1/\epsilon^2)} \right\}.$$

Then, the following holds with probability at least $2/3$: Let $\tilde{\boldsymbol{v}}^*$ and $\tilde{z}^*$ be an optimal solution and the optimal value, respectively, of the problem

$$\min_{\boldsymbol{v}\in\mathbb{R}^{|S|}} \psi_{|S|,A|_S,\boldsymbol{d}|_S,\boldsymbol{b}|_S}(\boldsymbol{v}) \ ,$$

where $\cdot|_S$ is an operator that extracts a submatrix (or subvector) specified by an index set S. Then,

$$\left|\frac{1}{|S|^2}\tilde{z}^* - \frac{1}{n^2}z^*\right| \le \epsilon L \max\left\{\frac{\|\tilde{\boldsymbol{v}}^*\|_2^2}{|S|}, \frac{\|\boldsymbol{v}^*\|_2^2}{n}\right\} \ ,$$

where $L = \max\{\max_{i,j}|A_{ij}|, \max_i|d_i|, \max_i|b_i|\}$.

Recently, Hayashi and Yoshida [10] proposed a constant-time sampling method for this problem with an additive error of $O(\epsilon L K_\infty^2 n^2)$ for $K_\infty = \max\{\|\boldsymbol{v}^*\|_\infty, \|\tilde{\boldsymbol{v}}^*\|_\infty\}$, where $\boldsymbol{v}^*$ and $\tilde{\boldsymbol{v}}^*$ are the optimal solutions to the original and sampled problems, respectively. Although their algorithm runs in constant time, the guarantee is not meaningful when $K_\infty = \omega(1)$ because the optimal value is always of order $O(Ln^2)$. Theorem 1 shows that we can improve

the additive error to $O(\epsilon L K_2^2 n^2)$, where $K_2 = \max\{\|\boldsymbol{v}^*\|_2/\sqrt{n}, \|\tilde{\boldsymbol{v}}^*\|_2/\sqrt{s}\}$, as long as the number of samples s is polylogarithmic (or more). We note that we always have $K_2 \leq K_\infty$ and the difference is significant when $\boldsymbol{v}^*$ and $\tilde{\boldsymbol{v}}^*$ are sparse. For example, if $\boldsymbol{v}^*$ and $\tilde{\boldsymbol{v}}^*$ have only $O(1)$ non-zero elements, then we have $K_2 = O(K_\infty/\sqrt{s})$. Our new bound provides a trade off between the additive error and the time complexity, which was unclear from the argument by Hayashi and Yoshida [10].

Moreover, we consider minimization over a sphere. Specifically, given a matrix $A \in \mathbb{R}^{n\times n}$, vectors $\boldsymbol{b}, \boldsymbol{d} \in \mathbb{R}^n$ and $r > 0$, we consider the following quadratic function minimization problem over a sphere of radius r:

$$\min_{\boldsymbol{v}:\|\boldsymbol{v}\|_2 \leq r} \psi_{n,A,\boldsymbol{d},\boldsymbol{b}}(\boldsymbol{v}) \ . \tag{2}$$

We give the first sublinear-time approximation algorithm for this problem.

▶ **Theorem 2.** *Let $\boldsymbol{v}^*$ and z^* be an optimal solution and optimal value, respectively, of Problem (2). Let $\epsilon > 0$ and let S be a random set such that each index $i \in \{1, 2, \ldots, n\}$ is taken to S independently w.p k/n with*

$$k = \max\left\{O\left(\frac{\log^2 n}{\epsilon^2}\right), \left(\frac{1}{\epsilon}\right)^{O(1/\epsilon^2)}\right\} .$$

Then, the following holds with probability at least $2/3$: Let $\tilde{\boldsymbol{v}}^*$ and $\tilde{z}^*$ be an optimal solution and the optimal value, respectively, of the problem

$$\min_{\|v\|_2 \leq \sqrt{\frac{|S|}{n}} r} \psi_{|S|,A|_S,d|_S,b|_S}(\boldsymbol{v}) \ .$$

Then,

$$\left|\frac{1}{|S|^2}\tilde{z}^* - \frac{1}{n^2}z^*\right| \leq \frac{\epsilon L r^2}{n} \ ,$$

where $L = \max\{\max_{i,j}|A_{ij}|, \max_i|d_i|, \max_i|b_i|\}$.

We can design a constant-time algorithm for (2) by using the result of [10], but the resulting error bound will be $O(\epsilon L r^2)$, which is n times worse than the bound in Theorem 2.

The proofs of Theorems 1 and 2 rely on a novel decomposition theorem of matrices, which will be discussed in Section 1.3. As another application of this decomposition theorem, we show that for any (small) t, we can approximate the t-th largest singular values of a matrix $A \in [-L, L]^{n\times m}$ (denoted $\sigma_t(A)$) to within an additive error of $O(L\sqrt{\epsilon t n m})$ in time

$$\max\left\{O\left(\frac{\max\{\log^2 n, \log^2 m\}}{\epsilon^2}\right), \left(\frac{1}{\epsilon}\right)^{O(1/\epsilon^2)}\right\} .$$

Our algorithms are very simple to implement, and do not require any structure in the input matrix. However, similar results (with better running time) can be obtained by applying known sampling techniques from [8]. Formally, we prove the following.

▶ **Theorem 3.** *Given a matrix $A \in [-L, L]^{n\times m}$, $\epsilon \in (0, 1)$, let*

$$k = \max\left\{O\left(\frac{\max\{\log^2 n, \log^2 m\}}{\epsilon^2}\right), \left(\frac{1}{\epsilon}\right)^{O(1/\epsilon^2)}\right\} .$$

Then, for every $t = O(k)$, there is an algorithm that runs in $\mathrm{poly}(k)$ time, and outputs a value z such that with probability at least $2/3$,

$$|\sigma_t(A) - z| \leq L\sqrt{\epsilon t n m}.$$

We note that since the $\sigma_t(A) \leq L\sqrt{\frac{nm}{t}}$ (see Fact 5), the *relative* error the algorithm achieves is at least $\sqrt{\epsilon} \cdot t$. Therefore, to get meaningful approximation, one must have that $\sqrt{\epsilon} \cdot t < 1$. So, if we wish to set $\epsilon = O(1)$ then we must have $t = O(1)$. We refer the reader to the full version for the singular values algorithms and their analysis.

1.2 Related work

In machine learning context, Clarkson *et al.* [5] considered several machine learning optimization problems and gave sublinear-time approximation algorithms for those problems. In particular, they considered approximate minimization of a quadratic function over the unit simplex $\Delta = \{\boldsymbol{x} \in \mathbb{R}^n \mid x_i \geq 0,\ \sum_i x_i = 1\}$. Namely, given a positive semidefinite matrix $A \in \mathbb{R}^{n\times n}$ and $\boldsymbol{b} \in \mathbb{R}^n$, they showed that it is possible to obtain an approximate solution to $\min_{x\in\Delta} \boldsymbol{x}^\top A\boldsymbol{x} + \boldsymbol{x}^\top \boldsymbol{b}$ (up to an additive error of ϵ) in $\widetilde{O}(n/\epsilon^2)$ time, which is sublinear in the input size $\Theta(n^2)$. In contrast, our algorithms run in polylogarithmic time and are much more efficient. Hayashi and Yoshida [11] proposed a constant-time approximation algorithm for Tucker decomposition of tensors, which can be seen as minimizing low-degree polynomials.

In addition to the work of Hayashi and Yoshida [10] mentioned above, an additional line of relevant work is constant-time approximation algorithms for the max cut problem on dense graphs [6, 16]. Let $L_G \in \mathbb{R}^{n\times n}$ be the Laplacian matrix of a graph G on n vertices. Then, the max cut problem can be seen as maximizing $\langle \boldsymbol{x}, L_G\boldsymbol{x}\rangle$ subject to $x_i \in \{-1/\sqrt{n}, 1/\sqrt{n}\}$, and these methods approximate the optimal value to within $O(\epsilon n)$. Our method for approximating the largest singular values can be seen as an extension of these methods to a continuous setting.

1.3 Techniques

The main ingredient in our proof is a novel *spectral decomposition theorem* of matrices, which may be of independent interest. The theorem states that we can decompose a matrix $A \in \mathbb{R}^{n\times m}$ into a structured matrix $A^{\mathrm{str}} \in \mathbb{R}^{n\times m}$ and a pseudorandom matrix $A^{\mathrm{psd}} \in \mathbb{R}^{n\times m}$. Here, A^{str} is structured in the sense that it is a block matrix with a polylogarithmic number of blocks such that the entries in each block are equal. Also, A^{psd} is pseudorandom in the sense that it has a small spectral norm. Formally, we prove the following. For a matrix $A \in \mathbb{R}^{n\times m}$, we define its max norm as $\|A\|_{\max} = \max_{1\leq i\leq n} \max_{1\leq j\leq m} |A_{ij}|$.

▶ **Theorem 4.** *For any matrix $A \in [-L, L]^{n\times m}$ and $\gamma \in (0, 1)$, there exists a decomposition $A = A^{str} + A^{psd}$ with the following properties for $N = \sqrt{nm}$:*

1. *A^{str} is* structured *in the sense that it is a block matrix with $\left(\frac{1}{\gamma}\right)^{O(1/\gamma^2)}$ blocks, such that the entries in each block are equal.*
2. *$\|A^{psd}\|_2 \leq \gamma NL$.*
3. *$\|A^{str}\|_{\max} = L/\gamma^{O(1)}$.*

Our decomposition theorem is a strengthening of the matrix decomposition result of Frieze and Kannan [7, 6]. In particular, they showed that any matrix $A \in \mathbb{R}^{n\times m}$ can be decomposed to $D_1 + \cdots + D_s + W$ for $s = O(1/\gamma^2)$, where the matrices D_i are block matrices and $\|W\|_C \leq \gamma nm\|A\|_{\max}$. Here, $\|W\|_C$ is the cut norm, which is defined as

$$\max_{S\subseteq\{1,\ldots,n\}} \max_{T\subseteq\{1,\ldots,m\}} \left|\sum_{i\in S}\sum_{j\in T} W_{ij}\right| .$$

By using a result of Nikiforov [20] that $\|W\|_2 = O(\sqrt{nm \cdot \|W\|_{\max} \cdot \|W\|_C})$ and the fact that the Frieze-Kannan result implies $\|W\|_{\max} \leq \sqrt{s}\|A\|_{\max}$, we get that $\|W\|_2 = O(nm \cdot \|A\|_{\max})$, which is too loose, and thus insufficient for our applications.

Given our decomposition theorem, we can conclude the following. When approximating (1) and (2), we can disregard the pseudorandom part A^{psd}. This will not affect our approximation by much, since A^{psd} has a small spectral norm. In addition, as A^{str} consists of a polylogarithmic number of blocks, such that the entries in each block are equal, we can hit all the blocks by sampling a polylogarithmic number of indices. Hence, we can expect that $A|_S$ is a good approximation to A. To formally define the distance between A and $A|_S$ and to show it is small, we exploit graph limit theory, initiated by Lovász and Szegedy [14] (refer to [13] for a book).

2 Preliminaries

For an integer n we let $[n] \stackrel{\text{def}}{=} \{1, 2, \ldots, n\}$. Given a set of indices $S = \{i_1, \ldots, i_k\}$, and a vector $\boldsymbol{v} \in \mathbb{R}^n$, we let $\boldsymbol{v}|_S \in \mathbb{R}^k$ be the *restriction* of $\boldsymbol{v}$ to S; that is, $(\boldsymbol{v}|_S)_j = v_{i_j}$, for every $i \in [k]$. Similarly, for a matrix $A \in \mathbb{R}^{n \times m}$ and sets $S_R = \{i_1, \ldots, i_{k_R}\} \subseteq [n]$ and $S_C = \{j_1, \ldots, j_{k_C}\} \subseteq [m]$, we denote the *restriction* of A to $S_R \times S_C$ by $A|_{S_R \times S_C} \in \mathbb{R}^{k_R \times k_C}$; that is, $(A|_{S_R \times S_C})_{j\ell} = A_{i_j i_\ell}$, for every $j \in [k_R]$ and $\ell \in [k_C]$. When $S_R = S_C = S$ we often use $A|_S$ as a shorthand for $A|_{S \times S}$. We use the notation $x = y \pm z$ as a shorthand for $y - z \leq x \leq y + z$.

Given a matrix $A \in \mathbb{R}^{n \times m}$ we define the *Frobenius norm* of A as $\|A\|_F = \sqrt{\sum_{(i,j) \in [n] \times [m]} A_{ij}^2}$ and the *max norm* of A as $\|A\|_{\max} = \max_{i \in [n], j \in [m]} |A_{ij}|$. For a matrix $A \in \mathbb{R}^{n \times m}$, we let $\sigma_\ell(A)$ denote the ℓ-th largest singular value of A. It is well known that the largest singular value can be evaluated using the following.

$$\sigma_1(A) = \max_{\boldsymbol{v} \in \mathbb{R}^m :\, \|\boldsymbol{v}\|_2 \leq 1} \|A\boldsymbol{v}\|_2 \,.$$

In addition, we state the following fact regarding the singular values.

▶ **Fact 5.** *Let $A \in \mathbb{R}^{n \times m}$, and consider the singular values of A: $\sigma_1(A) \geq \ldots \geq \sigma_{\min\{n,m\}}(A)$. Then, for every $1 \leq \ell \leq \min\{n, m\}$, $\sigma_\ell(A) \leq \frac{\|A\|_F}{\sqrt{\ell}}$.*

3 Spectral Decomposition Theorem

In this section we will prove the following decomposition theorem.

▶ **Theorem 6** (Spectral decomposition, restatement of Theorem 4)**.** *For any matrix $A \in [-L, L]^{n \times m}$ and $\gamma \in (0, 1)$, there exists a decomposition $A = A^{\text{str}} + A^{\text{psd}}$ with the following properties for $N = \sqrt{nm}$:*

1. *A^{str} is* structured *in the sense that it is a block matrix with $O\left(\left(\frac{1}{\gamma^{10}}\right)^{3/\gamma^2}\right)$ blocks, such that the entries in each block are equal.*
2. *$\|A^{\text{psd}}\|_2 \leq 7\gamma NL$.*
3. *$\|A^{\text{str}}\|_{\max} \leq \frac{2}{\gamma^{11}} L$.*

The above theorem will serve as a central tool in the analysis of our algorithms. The fact that A^{str} is a block matrix with polylogarithmic number of blocks, such that the entries in each block are equal, implies that by using polylogarithmic number of samples, we can query

(with high probability) an entry from each of the blocks. In addition, the fact that A^{psd} has a small spectral norm allows us to disregard it, which only paying a small cost in the error of our approximation.

In order to prove the theorem, we introduce the following definition, two lemmas and a claim.

▶ **Definition 7.** We say that a partition $\mathcal{Q}$ is a *refinement* of a partition $\mathcal{P} = \{V_1, \ldots, V_p\}$, if $\mathcal{Q}$ is obtained from $\mathcal{P}$, by splitting some sets V_i into one or more parts.

▶ **Lemma 8.** *Given a matrix $A \in [-L, L]^{n\times m}$ and $\gamma \in (0, 1)$, there exists a block matrix $A^{\text{str}} \in \mathbb{R}^{n\times m}$ with $O\left(\left(\frac{1}{\gamma^{10}}\right)^{3/\gamma^2}\right)$ blocks such that the entries in each block are equal and $\|A - A^{\text{str}}\|_2 \le 7\gamma NL$, where $N = \sqrt{nm}$.*

In order to prove Lemma 8, we will need to prove the following.

▶ **Lemma 9.** *Given a matrix $A \in [-L, L]^{n\times m}$ and $\gamma \in (0, 1)$, let $A' = \sum_{\ell:\sigma_\ell \ge \gamma NL} \sigma_\ell \boldsymbol{u}^\ell (\boldsymbol{v}^\ell)^\top$. Then, $\|A'\|_{\max} \le \frac{L}{\gamma^3}$.*

Proof. Assume that A has s singular values such that $\sigma_\ell \ge \gamma NL$. For any $\ell \in [s]$, let $M_\ell = \sigma_\ell \boldsymbol{u}^\ell (\boldsymbol{v}^\ell)^\top$, and let B denote $\sum_{\ell:\ \sigma_\ell < \gamma NL} \sigma_\ell \boldsymbol{u}^\ell (\boldsymbol{v}^\ell)^\top$. Then, we can write A as,

$$A = \underbrace{\sigma_1 \boldsymbol{u}^1 (\boldsymbol{v}^1)^\top + \cdots + \sigma_s \boldsymbol{u}^s (\boldsymbol{v}^s)^\top}_{A'} + \sum_{\ell:\ \sigma_\ell < \gamma NL} \sigma_\ell \boldsymbol{u}^\ell (\boldsymbol{v}^\ell)^\top = M_1 + \cdots + M_s + B \ .$$

Consider any $\ell \in [s]$, $(i, j) \in [n] \times [m]$, and let $\beta_{ij}^\ell \stackrel{\text{def}}{=} |\sigma_\ell u_i^\ell v_j^\ell|$.

Let c_j^A denote the j-th column of the matrix A. So, $c_j^A = c_j^{M_1} + \cdots + c_j^{M_s} + c_j^B$. For any $\ell \in [s]$, $c_j^{M_\ell}$ is perpendicular to c_j^B and all $\{c_j^{M_t}\}_t$ such that $t \ne \ell$, and thus,

$$\|c_j^A\|_2 \ge \|c_j^{M_\ell}\|_2 = \|\sigma_\ell \boldsymbol{u}^\ell v_j^\ell\|_2 = \sigma_\ell |v_j^\ell| \ .$$

Let r_i^A denote the i-th row of the matrix A. Then similarly, we have $\|r_i^A\|_2 \ge \sigma_\ell |u_i^\ell|$, and it follows that $\|c_j^A\|_2 \|r_i^A\|_2 \ge \sigma_\ell \beta_{ij}^\ell$.

On the other hand, since $A \in [-L, L]^{n\times m}$, we have $\|c_j^A\|_2 \le \sqrt{n}L$ and $\|r_i^A\|_2 \le \sqrt{m}L$, and therefore,

$$\beta_{ij}^\ell \le \frac{\sqrt{nm}L^2}{\sigma_\ell} \le \frac{\sqrt{nm}L^2}{\gamma NL} = \frac{L}{\gamma} \ .$$

By the fact that $s \le \frac{1}{\gamma^2}$, we get that $\|A'\|_{\max} \le \frac{L}{\gamma^3}$, which concludes the proof. ◀

With this lemma at hand, we are ready to prove Lemma 8.

Proof of Lemma 8: Recall that A can be written as,

$$A = \sum_\ell \sigma_\ell \boldsymbol{u}^\ell (\boldsymbol{v}^\ell)^\top \ ,$$

where $\sigma_1 \ge \cdots \ge \sigma_{\min\{n,m\}} \ge 0$ are the singular values of A and $\boldsymbol{u}^1, \ldots, \boldsymbol{u}^{\min\{n,m\}} \in \mathbb{R}^n$ and $\boldsymbol{v}^1, \ldots, \boldsymbol{v}^{\min\{n,m\}} \in \mathbb{R}^m$ are the corresponding left and right singular vectors. If we let A' be such that

$$A' = \sum_{\ell:\ \sigma_\ell \ge \gamma NL} \sigma_\ell \boldsymbol{u}^\ell (\boldsymbol{v}^\ell)^\top \ ,$$

then we have that $\frac{\|(A-A')\boldsymbol{x}\|_2}{\|x\|_2} \leq \gamma NL$ for any $\boldsymbol{x} \in \mathbb{R}^m$.

Next we show the existence of A^{str}, which is a block matrix (with $O\left(\left(\frac{1}{\gamma^{10}}\right)^{3/\gamma^2}\right)$ blocks), such that A^{str} has the same value on every block. We construct A^{str} as follows.

Let $\epsilon = \epsilon(\gamma)$ be determined later and let $J = \frac{1}{\gamma^2}$. Let $\delta_n \stackrel{\mathrm{def}}{=} \frac{\epsilon}{J\sqrt{n}}$ and $T \stackrel{\mathrm{def}}{=} (\frac{J}{\epsilon})^{3/2}$, and partition the interval $[0,1]$ into buckets $B_1^n, \ldots, B_T^n, B_{\mathrm{Large}}^n$ such that

$$B_t^n \stackrel{\mathrm{def}}{=} [(t-1)\delta_n, t\delta_n) \;\; \forall t \in [T] \qquad \text{and} \qquad B_{\mathrm{Large}}^n \stackrel{\mathrm{def}}{=} [\sqrt{J/\epsilon n}, 1] \,.$$

For every $\boldsymbol{u}^\ell$ such that $\sigma_\ell \geq \gamma NL$, we define a partition $\mathcal{P}_R^\ell = \{P_{R,1}^\ell \ldots, P_{R,T}^\ell, \Sigma_R^\ell\}$ of the indices in $[n]$ so that $P_{R,t}^\ell = \{i \in [n] \mid |u_i^\ell| \in B_t^n\}$ for each $t \in [T]$ and $\Sigma_R^\ell = \{i \in [n] \mid |u_i^\ell| \in B_{\mathrm{Large}}^n\}$. We eliminate emptysets from $\mathcal{P}_R^\ell$ if exist. Note that by the definition of Σ_R^ℓ we have that $|\Sigma_R^\ell| \leq \epsilon n/J$. Next, for every ℓ such that $\sigma_\ell \geq \gamma NL$, we define $\hat{\boldsymbol{u}}^\ell$ as follows.

$$\hat{u}_i^\ell = \begin{cases} 0, & \text{if } |u_i^\ell| \in B_{\mathrm{Large}}^n \,, \\ \delta_n(t-1), & \text{if } |u_i^\ell| \in B_t^n \text{ for some } t \in [T] \,. \end{cases} .$$

Next, let $\Sigma_R = \bigcup_{\ell:\sigma_\ell \geq \gamma NL} \Sigma_R^\ell$ and let $\mathcal{P}_R$ be a partition of $[n] \setminus \Sigma_R$ that refines all $\{P_{R,t}^\ell \mid P_{R,t}^\ell \neq \emptyset\}$ for which ℓ is such that $\sigma_\ell \geq \gamma NL$.

Similarly define $\hat{\boldsymbol{v}}^\ell$ from $\boldsymbol{v}^\ell$ by setting $\delta_m = \frac{\epsilon}{J\sqrt{m}}$, and defining $\mathcal{P}_C^\ell = \{P_{C,1}^\ell, \ldots, P_{C,T}^\ell, \Sigma_C^\ell\}$ analogously for each ℓ. Let $\Sigma_C = \bigcup_{\ell:\sigma_\ell \geq \gamma NL} \Sigma_C^\ell$ and let $\mathcal{P}_C$ be a partition of $[m] \setminus \Sigma_C$ that refines all $\{P_{C,t}^\ell \mid P_{C,t}^\ell \neq \emptyset\}$ for which ℓ is such that $\sigma_\ell \geq \gamma NL$.

Define,

$$A^{\mathrm{str}} \stackrel{\mathrm{def}}{=} \sum_{\ell:\, \sigma_\ell \geq \gamma NL} \sigma_\ell \hat{\boldsymbol{u}}^\ell (\hat{\boldsymbol{v}}^\ell)^\top \qquad \text{and} \qquad A^{\mathrm{psd}} \stackrel{\mathrm{def}}{=} A - A^{\mathrm{str}}.$$

By Fact 5 we have that $\sigma_\ell \leq \frac{NL}{\sqrt{\ell}}$ for all $\ell \geq 1$. Therefore, we can have at most $J = 1/\gamma^2$ indices such that $\sigma_\ell \geq \gamma NL$. Thus, the partition $\mathcal{P}_R$ satisfies $|\mathcal{P}_R| \leq \left(\left(\frac{J}{\epsilon}\right)^{\frac{3}{2\gamma^2}}\right) = \left(\frac{1}{\gamma^2\epsilon}\right)^{(3/2\gamma^2)}$. Similarly, we have $|\mathcal{P}_C| \leq \left(\frac{1}{\gamma^2\epsilon}\right)^{(3/2\gamma^2)}$. Therefore, the resulting matrix is a block matrix with $O\left(\left(\frac{1}{\gamma^2\epsilon}\right)^{\frac{3}{\gamma^2}}\right)$ many blocks, such that all the entries in each block are the same. We refer to $\mathcal{P}_R$ and $\mathcal{P}_C$ the *row partition of* A^{str} and the *column partition of* A^{str} respectively.

Next, we have that

$$\begin{aligned} \|A^{\mathrm{psd}}\boldsymbol{x}\| = \|(A - A^{\mathrm{str}})\boldsymbol{x}\|_2^2 &\leq \|(A-A')\boldsymbol{x}\|_2^2 + \|(A' - A^{\mathrm{str}})\boldsymbol{x}\|_2^2 \\ &\leq \gamma^2 N^2 L^2 \|\boldsymbol{x}\|_2^2 + \|(A' - A^{\mathrm{str}})\boldsymbol{x}\|_2^2 \,. \end{aligned}$$

Consider the "error" term $\|(A' - A^{\mathrm{str}})\boldsymbol{x}\|_2^2$.

$$\begin{aligned} \|(A' - A^{\mathrm{str}})\boldsymbol{x}\|_2^2 \leq \|\boldsymbol{x}\|_2^2 \sum_{(i,j)\in[n]\times[m]} \left(A'_{ij} - A^{\mathrm{str}}_{ij}\right)^2 = \|\boldsymbol{x}\|_2^2 \Bigg(&\sum_{(i,j)\in\bar{\Sigma}_R\times\bar{\Sigma}_C} \left(A'_{ij} - A^{\mathrm{str}}_{ij}\right)^2 \\ + &\sum_{(i,j)\in(\Sigma_R\times\bar{\Sigma}_C)\cup(\bar{\Sigma}_R\times\Sigma_C)} \left(A'_{ij} - A^{\mathrm{str}}_{ij}\right)^2 + \sum_{(i,j)\in\Sigma_R\times\Sigma_C} \left(A'_{ij} - A^{\mathrm{str}}_{ij}\right)^2 \Bigg), \end{aligned}$$

where $\bar{\Sigma}_R = [n] \setminus \Sigma_R$ and $\bar{\Sigma}_C = [m] \setminus \Sigma_C$.

We analyze each of these terms separately. First, note that

$$\left(A'_{ij} - A^{\mathrm{str}}_{ij}\right)^2 = \left(\sum_{\ell:\sigma_\ell \geq \gamma NL} \sigma_\ell (u_i^\ell v_j^\ell - \hat{u}_i^\ell \hat{v}_j^\ell)\right)^2 .$$

So,

$$\begin{aligned}\sum_{(i,j)\in \bar{\Sigma}_R \times \bar{\Sigma}_C} \left(A'_{ij} - A^{\mathrm{str}}_{ij}\right)^2 &= \sum_{(i,j)\in \bar{\Sigma}_R \times \bar{\Sigma}_C} \left(\sum_{\ell:\sigma_\ell \geq \gamma NL} \sigma_\ell (u_i^\ell v_j^\ell - \hat{u}_i^\ell \hat{v}_j^\ell)\right)^2 \\ &\overset{(*)}{\leq} \sum_{(i,j)\in \bar{\Sigma}_R \times \bar{\Sigma}_C} \left(\frac{2\sqrt{\epsilon}}{N} \sum_{\ell:\sigma_\ell \geq \gamma NL} \sigma_\ell\right)^2 \\ &\overset{(**)}{\leq} \sum_{(i,j)\in \bar{\Sigma}_R \times \bar{\Sigma}_C} \left(\frac{2\sqrt{\epsilon}}{N} \cdot \frac{2NL}{\gamma}\right)^2 \leq \frac{16\epsilon N^2 L^2}{\gamma^2} .\end{aligned}$$

Here, $(*)$ follows from the fact that for two indices $i \in \bar{\Sigma}_R, j \in \bar{\Sigma}_C$ we have that $\hat{u}_i^\ell \hat{v}_j^\ell = (u_i^\ell \pm \delta_n)(v_j^\ell \pm \delta_m) \leq u_i^\ell v_j^\ell \pm \frac{2\sqrt{\epsilon}}{N}$. $(**)$ follows from the fact that there can be at most $1/\gamma^2$ indices ℓ, such that $\sigma_\ell \geq \gamma NL$, and therefore

$$\sum_{\ell:\sigma_\ell \geq \gamma NL} \sigma_\ell \leq \sum_{\ell=1}^{1/\gamma^2} \sigma_\ell \leq \sum_{\ell=1}^{1/\gamma^2} \frac{NL}{\sqrt{\ell}} \leq \frac{2NL}{\gamma} . \tag{3}$$

Next, we have that,

$$\begin{aligned}&\sum_{(i,j)\in(\Sigma_R \times \bar{\Sigma}_C)\cup(\bar{\Sigma}_R \times \Sigma_C)} \left(A'_{ij} - A^{\mathrm{str}}_{ij}\right)^2 \\ &= \sum_{(i,j)\in(\Sigma_R \times \bar{\Sigma}_C)\cup(\bar{\Sigma}_R \times \Sigma_C)} \left(\sum_{\ell:\sigma_\ell \geq \gamma NL} \sigma_\ell (u_i^\ell v_j^\ell - \hat{u}_i^\ell \hat{v}_j^\ell)\right)^2 \\ &\overset{(***)}{=} \sum_{(i,j)\in(\Sigma_R \times \bar{\Sigma}_C)\cup(\bar{\Sigma}_R \times \Sigma_C)} \left(A'_{ij}\right)^2 \leq \sum_{(i,j)\in(\Sigma_R \times \bar{\Sigma}_C)\cup(\bar{\Sigma}_R \times \Sigma_C)} \left(\frac{L}{\gamma^3}\right)^2 \leq \frac{2\epsilon N^2 L^2}{\gamma^6} .\end{aligned}$$

Here, $(***)$ follows from the fact that when one of the indices is in Σ_R or Σ_C we set the corresponding entry in the rounded vector to 0. In addition, the last inequality follows from the fact that when we remove the lower singular part of the matrix, we can only increase the value by at most factor of $1/\gamma^3$ (see Lemma 9). Finally,

$$\begin{aligned}\sum_{(i,j)\in \Sigma_R \times \Sigma_C} \left(A'_{ij} - A^{\mathrm{str}}_{ij}\right)^2 &= \sum_{(i,j)\in \Sigma_R \times \Sigma_C} \left(\sum_{\ell:\sigma_\ell \geq \gamma NL} \sigma_\ell (u_i^\ell v_j^\ell - \hat{u}_i^\ell \hat{v}_j^\ell)\right)^2 \\ &= \sum_{(i,j)\in \Sigma_R \times \Sigma_C} \left(A'_{ij}\right)^2 \leq \frac{\epsilon^2 N^2 L^2}{\gamma^6}\end{aligned}$$

Combining all the three terms and setting $\epsilon = \gamma^8$ gives,

$$\|(A' - A^{\mathrm{str}})\boldsymbol{x}\|_2^2 \leq \|\boldsymbol{x}\|_2^2 \left(\frac{16\epsilon}{\gamma^2} + \frac{2\epsilon}{\gamma^6} + \frac{\epsilon^2}{\gamma^6}\right) N^2 L^2 \leq 19\gamma^2 \|\boldsymbol{x}\|_2^2 N^2 L^2.$$

Therefore, we get that,

$$\begin{aligned}\|(A-A^{\mathrm{str}})\boldsymbol{x}\|_2^2 &\le 2\|(A-A')\boldsymbol{x}\|_2^2+2\|(A'-A^{\mathrm{str}})\boldsymbol{x}\|_2^2\\ &\le 2(\gamma^2N^2L^2\|\boldsymbol{x}\|_2^2+19\gamma^2\|\boldsymbol{x}\|_2^2N^2L^2)\le 40\gamma^2\|\boldsymbol{x}\|_2^2N^2L^2\ ,\end{aligned}$$

and the lemma follows. ◀

We are left with bounding the max norm of A^{str}.

▶ **Claim 10.** *Given a matrix $A\in[-L,L]^{n\times m}$ and $\gamma\in(0,1)$, let $A^{\mathrm{str}}\in\mathbb{R}^{n\times m}$ be the block approximation matrix defined above. Then, $\|A^{\mathrm{str}}\|_{\max}\le\frac{2L}{\gamma^{11}}$*

Proof. By the definition of A^{str} we have that $|A^{\mathrm{str}}_{ij}|=|\sum_{\ell:\sigma_\ell\ge\gamma NL}\sigma_\ell\hat{u}_i^\ell\hat{v}_j^\ell|$. By the definition of the rounding process, we have that for every $i\in[n]$ we have that $|\hat{u}_i^\ell|\le\sqrt{\frac{J}{\epsilon n}}=\frac{1}{\gamma^5\sqrt{n}}$ (recall that $J=1/\gamma^2$ and $\epsilon=\gamma^8$). Similarly, for every $j\in[m]$ we have that $|\hat{v}_j^\ell|\le\frac{1}{\gamma^5\sqrt{m}}$. Therefore,

$$|A^{\mathrm{str}}_{ij}|=\left|\sum_{\ell:\sigma_\ell\ge\gamma NL}\sigma_\ell\hat{u}_i^\ell\hat{v}_j^\ell\right|\le\frac{1}{\gamma^{10}N}\sum_{\ell:\sigma_\ell\ge\gamma NL}\sigma_\ell\le\frac{2L}{\gamma^{11}}\ ,$$

where the last inequality uses (3). ◀

Proof of Theorem 6: The proof follows directly from Lemma 8 and Claim 10. ◀

4 Dikernels and Sampling Lemmas

In this section we will formalize the idea that $A|_{S_R\times S_C}$ is a good approximation of A when S_R and S_C are uniformly random subsets of indices. (The proof for $A|_S$, where S is uniformly random subset of indices, is almost identical and we omit it.) We start by providing some background on dikernels and their connection to matrices and then move on to proving our sampling lemmas.

4.1 Dikernels and Matrices

We call a (measurable) function $f\colon[0,1]^2\to\mathbb{R}$ a *dikernel*. We can regard a dikernel as a matrix whose index is specified by a real value in $[0,1]$. For two functions $f,g:[0,1]\to\mathbb{R}$, we define their inner product as $\langle f,g\rangle\stackrel{\mathrm{def}}{=}\int_0^1 f(x)g(x)dx$. For a dikernel $\mathcal{A}:[0,1]^2\to\mathbb{R}$ and a function $f:[0,1]\to\mathbb{R}$, we define the function $\mathcal{A}f\colon[0,1]\to\mathbb{R}$ as $(\mathcal{A}f)(x)=\langle\mathcal{A}(x,\cdot),f\rangle$. In addition, we define the *spectral norm* of $\mathcal{A}$ as $\|\mathcal{A}\|_2\stackrel{\mathrm{def}}{=}\sup_{f\colon[0,1]\to\mathbb{R}}\frac{\|\mathcal{A}f\|_2}{\|f\|_2}$, and the *Frobenius norm* of $\mathcal{A}$ as $\|\mathcal{A}\|_F\stackrel{\mathrm{def}}{=}\sqrt{\int_0^1\int_0^1\mathcal{A}(x,y)^2dxdy}$.

For an integer $n\in\mathbb{N}$, let $I_1^n=[0,\frac{1}{n}]$, and for every $1<k\le n$, let $I_k^n=(\frac{k-1}{n},\frac{k}{n}]$. For $x\in[0,1]$, we define $i^n(x)$ as the unique integer $k\in[n]$ such that $x\in I_k^n$.

▶ **Definition 11.** Given a matrix $A\in\mathbb{R}^{n\times m}$, we construct the corresponding dikernel $\mathcal{A}$ as $\mathcal{A}(x,y)=A_{i^n(x),i^m(y)}$. In addition, given two sets of indices $S_R\subseteq[n]$ and $S_C\subseteq[m]$, when we write $\mathcal{A}|_{\mathcal{S}_R\times\mathcal{S}_C}$, we first extract the matrix $A|_{S_R\times S_C}$ and then consider its corresponding dikernel.

The following lemma shows that the spectral norms of A and $\mathcal{A}$ are essentially the same up to normalization. The proof can be found in Appendix A.1.

▶ **Lemma 12.** *Let $A \in \mathbb{R}^{n\times m}$ be a matrix. Then, we have*

$$\max_{\boldsymbol{v}\in\mathbb{R}^m} \frac{\|A\boldsymbol{v}\|_2^2}{\|\boldsymbol{v}\|_2^2} = nm \cdot \sup_{f\colon [0,1]\to\mathbb{R}} \frac{\|\mathcal{A}f\|_2^2}{\|f\|_2^2} \ .$$

▶ **Corollary 13.** *Let $A \in \mathbb{R}^{n\times m}$ be a matrix. Then,*

$$\|A\|_2 = \sqrt{nm} \cdot \|\mathcal{A}\|_2 \ .$$

Proof. The proof is immediate by the definition of the spectral norm and Lemma 12. ◀

▶ **Definition 14.** Let μ be a Lebesgue measure. A map $\pi\colon [0,1] \to [0,1]$ is *measure preserving* if the pre-image $\pi^{-1}(X)$ is measurable for every measurable set X and $\mu(\pi^{-1}(X)) = \mu(X)$. A *measure preserving bijection* is a measure preserving map whose inverse map exists and is also measurable. For a measure preserving bijection π and a dikernel $\mathcal{A}$, we define the dikernel $\pi(\mathcal{A})$, as $\pi(\mathcal{A})(x,y) = \mathcal{A}(\pi(x),\pi(y))$.

4.2 Sampling Lemmas

In this subsection, we will prove that given matrices $A_1,\ldots,A_T \in [-L,L]^{n\times m}$, we obtain a good approximation of their corresponding dikernels, by sampling a small number of elements. The next lemma states that there is a way to "align" the sampled matrices with the original matrices. We refer the reader to Appendix A.1 for the full proofs.

▶ **Lemma 15.** *Given matrices $A_1,\ldots,A_T \in [-L,L]^{n\times n}$ and $\gamma \in (0,1)$, let $A_1^{str},\ldots,A_T^{str}$ be the block approximation matrices as in Lemma 8. In addition, for $t\in[T]$, let $\mathcal{P}_R^{A_t} = \{V_1^{A_t},\ldots,V_p^{A_t}\}$ and $\mathcal{P}_C^{A_t} = \{V_1'^{A_t},\ldots,V_q'^{A_t}\}$ be the row and column partitions of A_t^{str} (from Lemma 8). Let S_R be a set of size s_R, generated by picking each element in $[n]$ independently with probability k_R/n, and let S_C be a set of size s_C, generated by picking each element in $[m]$ independently with probability k_C/m for some $k_R,k_C>0$.*

Then, there exists a measure preserving bijection $\pi\colon[0,1]\to[0,1]$ such that for every $t\in[T]$

$$\mathop{\mathbf{E}}_{S_R,S_C}\left[\|\mathcal{A}_t^{str} - \pi(\mathcal{A}_t^{str}|_{S_R\times S_C})\|_2\right] = O\left(\frac{L}{\gamma^{11}}\cdot\max\left(\sqrt{\frac{p^{T/2}}{s_R^{1/2}}},\sqrt{\frac{q^{T/2}}{s_C^{1/2}}}\right)\right) \ .$$

In the following lemma we prove concentration around the mean.

▶ **Lemma 16.** *Let $\gamma>0$, and $A_1,\ldots,A_T\in[-L,L]^{n\times m}$. Let S_R be a set generated by picking each element in $[n]$ independently with probability k_R/n and let S_C be a set generated by picking each element in $[m]$ independently with probability k_C/m for some $k_R,k_C>0$. Then, with probability at least $89/100$ there exists a measure preserving bijection $\pi\colon[0,1]\to[0,1]$ such that for every $t\in[T]$,*

$$\|\mathcal{A}_t - \pi\left(\mathcal{A}_t|_{S_R\times S_C}\right)\|_2 \le 210\gamma LT + 10LT\left(\sqrt{\frac{8\log n}{k_C}} + \sqrt{\frac{8\log m}{k_R}} + \sqrt{\frac{4\log n\log m}{k_Rk_C}}\right) + O\left(\frac{LT}{\gamma^{11}}\left(\frac{1}{\gamma^{10}}\right)^{\frac{3T}{4\gamma^2}}\cdot\max\left(\frac{1}{k_R^{1/4}},\frac{1}{k_C^{1/4}}\right)\right),$$

where $N=\sqrt{nm}$.

Algorithm 1 Minimization Algorithm(A, n, ϵ, k).

1: Let $S \subseteq \{1, 2, \ldots, n\}$ such that each index i is taken to S independently w.p k/n.
2: **if** $|S| > 2k$ **then**
3: **Abort**
4: **return** $\min_{\boldsymbol{v} \in \mathbb{R}^{|S|}} \psi_{|S|, A|_S, \boldsymbol{d}|_S, \boldsymbol{b}|_S}(\boldsymbol{v})$.

5 Applications

5.1 Quadratic Function Minimization

In this section, we show that we can approximately solve quadratic function minimization problems in polylogarithmic time.

Recall that we are given a matrix $A \in \mathbb{R}^{n \times n}$ and vectors $\boldsymbol{d}, \boldsymbol{b} \in \mathbb{R}^n$, and consider the following quadratic function minimization problem:

$$\min_{\boldsymbol{v} \in \mathbb{R}^n} \psi_{n,A,\boldsymbol{d},\boldsymbol{b}}(\boldsymbol{v}), \text{ where } \psi_{n,A,\boldsymbol{d},\boldsymbol{b}}(\boldsymbol{v}) = \langle \boldsymbol{v}, A\boldsymbol{v} \rangle + n \langle \boldsymbol{v}, \mathrm{diag}(\boldsymbol{d})\boldsymbol{v} \rangle + n \langle \boldsymbol{b}, \boldsymbol{v} \rangle \ . \tag{4}$$

Here $\mathrm{diag}(\boldsymbol{d}) \in \mathbb{R}^{n \times n}$ is a matrix whose diagonal entries are specified by $\boldsymbol{d}$.

First, we describe our algorithm for minimizing quadratic functions. We first sample a set of indices $S \subseteq \{1, 2, \ldots, n\}$ with each index included with probability k/n, where k is some constant. If $|S|$ is too large, we immediately stop the process by claiming that the algorithm has failed. Otherwise, we solve the problem on $A|_S$, $\boldsymbol{d}|_S$, $\boldsymbol{b}|_S$ and then output the optimal solution. The detail is given in Algorithm 1.

Due to our extensive use of dikernels in the analysis, we introduce a continuous version of problem (4). The real valued function $\Psi_{n,A,\boldsymbol{d},\boldsymbol{b}}$ on function $f \colon [0,1] \to \mathbb{R}$ is defined as

$$\Psi_{n,A,\boldsymbol{d},\boldsymbol{b}}(f) = \langle f, \mathcal{A}f \rangle + \left\langle f^2, \mathcal{D}\mathbf{1} \right\rangle + \langle f, \mathcal{B}\mathbf{1} \rangle \ ,$$

where $\mathcal{D}$ and $\mathcal{B}$ are the corresponding dikernels of $\boldsymbol{d} \cdot \mathbf{1}^\top$ and $\boldsymbol{b} \cdot \mathbf{1}^\top$ respectively, $f^2 \colon [0,1] \to \mathbb{R}$ is a function such that $f^2(x) = f(x)^2$ for every $x \in [0,1]$ and $\mathbf{1} \colon [0,1] \to \mathbb{R}$ is the constant function that has the value 1 everywhere.

In order to prove Theorem 1, we prove that the minimizations of $\psi_{n,A,\boldsymbol{d},\boldsymbol{b}}$ and $\Psi_{n,A,\boldsymbol{d},\boldsymbol{b}}$ are essentially equivalent. The proof of the lemma can be found in Appendix A.2.

▶ **Lemma 17.** *Let $A \in \mathbb{R}^{n \times n}$ and $\boldsymbol{b}, \boldsymbol{d} \in \mathbb{R}^n$. Then, for any $r > 0$*

$$\min_{\boldsymbol{v} : \|\boldsymbol{v}\|_2 \leq r} \psi_{n,A,\boldsymbol{d},\boldsymbol{b}}(\boldsymbol{v}) = n^2 \cdot \inf_{f : \|f\|_2 \leq \frac{r}{\sqrt{n}}} \Psi_{n,A,\boldsymbol{d},\boldsymbol{b}}(f) \ .$$

With the above lemma, we are ready to prove our main result.

Proof of Theorem 1: By applying Chernoff bounds, we have that with probability at least $1 - o(1)$, the size of S is at most $2k$. As before, we apply Lemma 16 with $\gamma = O(\epsilon)$,

$$k = \max \left\{ O\left(\frac{\log^2 n}{\epsilon^2} \right), \left(\frac{1}{\epsilon} \right)^{O(1/\epsilon^2)} \right\} ,$$

and $S_C = S_R = S$[3]. Then, with probability at least $2/3$ there exists a measure preserving bijection $\pi \colon [0,1] \to [0,1]$ such that for any function $f \colon [0,1] \to \mathbb{R}$,

$$\max \left\{ \left| \langle f, (\mathcal{A} - \pi(\mathcal{A}|_S))f \rangle \right|, \left| \langle f, (\mathcal{D} - \pi(\mathcal{D}|_S))f \rangle \right|, \left| \langle f, (\mathcal{B} - \pi(\mathcal{B}|_S))f \rangle \right| \right\} \leq \frac{\epsilon L}{3} \|f\|_2^2 \ .$$

[3] We note that in this case, the sets S_R and S_C are dependent. However, the proof of Lemma 16 can be easily modified to the case where the matrix is in $[-L, L]^{n \times n}$, where for this case $S_R = S_C = S$.

Algorithm 2 Minimization Algorithm Over a Ball(A, n, ϵ, k, r).

1: Let $S \subseteq \{1, 2, \ldots, n\}$ such that each index i is taken to S independently w.p k/n.
2: **if** $|S| > 2k$ **then**
3: **Abort**
4: **return** $\min_{\|v\|_2 \leq \sqrt{\frac{|S|}{n}} r} \psi_{|S|, A|_S, \boldsymbol{d}|_S, \boldsymbol{b}|_S}(\boldsymbol{v})$.

Let $R = \max\left\{ \frac{\|\tilde{\boldsymbol{v}}^*\|_2}{\sqrt{|S|}}, \frac{\|\boldsymbol{v}^*\|_2}{\sqrt{n}} \right\}$. Then, by using Lemma 17:

$$
\begin{aligned}
\tilde{z}^* &= \min_{\boldsymbol{v} \in \mathbb{R}^{|S|}} \psi_{|S|, A|_S, \boldsymbol{d}|_S, \boldsymbol{b}|_S}(\boldsymbol{v}) = \min_{\boldsymbol{v}: \|\boldsymbol{v}\|_2 \leq R\sqrt{|S|}} \psi_{|S|, A|_S, \boldsymbol{d}|_S, \boldsymbol{b}|_S}(\boldsymbol{v}) \\
&= |S|^2 \cdot \min_{f: \|f\|_2 \leq R} \Psi_{|S|, A|_S, \boldsymbol{d}|_S, \boldsymbol{b}|_S}(f) \\
&= |S|^2 \cdot \min_{f: \|f\|_2 \leq R} \Big\{ \langle f, (\pi(\mathcal{A}|_{\mathcal{S}}) - \mathcal{A})\, f \rangle + \langle f, \mathcal{A} f \rangle + \left\langle f^2, (\pi(\mathcal{D}|_{\mathcal{S}}) - \mathcal{D})\, \mathbf{1} \right\rangle + \left\langle f^2, \mathcal{D}\mathbf{1} \right\rangle \\
&\qquad\qquad + \langle f, (\pi(\mathcal{B}|_{\mathcal{S}}) - \mathcal{B})\, \mathbf{1} \rangle + \langle f, \mathcal{B}\mathbf{1} \rangle \Big\} \\
&\leq |S|^2 \cdot \min_{f: \|f\|_2 \leq R} \left\{ \langle f, \mathcal{A} f \rangle + \left\langle f^2, \mathcal{D}\mathbf{1} \right\rangle + \langle f, \mathcal{B}\mathbf{1} \rangle \pm \epsilon L \|f\|_2^2 \right\} \\
&\leq |S|^2 \cdot \min_{f: \|f\|_2 \leq R} \Psi_{n, A, \boldsymbol{d}, \boldsymbol{b}}(f) \pm \epsilon L |S|^2 R^2 \\
&= \frac{|S|^2}{n^2} \cdot \min_{\boldsymbol{v}: \|\boldsymbol{v}\|_2 \leq \sqrt{n} R} \psi_{n, A, \boldsymbol{d}, \boldsymbol{b}}(\boldsymbol{v}) \pm \epsilon L |S|^2 R^2 \\
&= \frac{|S|^2}{n^2} \cdot \min_{\boldsymbol{v} \in \mathbb{R}^n} \psi_{n, A, \boldsymbol{d}, \boldsymbol{b}}(\boldsymbol{v}) \pm \epsilon L |S|^2 R^2 = \frac{|S|^2}{n^2} z^* \pm \epsilon L |S|^2 R^2 \,.
\end{aligned}
$$

By rearranging the inequality and applying the union bound the theorem follows. ◀

As a corollary we show that we can obtain an approximation algorithm for minimizing a quadratic function over a ball of radius r with better error bounds compared than the one obtained by Hayashi and Yoshida [10]. The proof of correctness is similar to the proof of Theorem 1.

▶ **Corollary 18** (Restatement of Theorem 2). *Let $\boldsymbol{v}^*$ and z^* be an optimal solution and optimal value, respectively, of $\min_{\|\boldsymbol{v}\|_2 \leq r} \psi_{n, A, \boldsymbol{d}, \boldsymbol{b}}(\boldsymbol{v})$. Let $\epsilon > 0$ and let S be a random set generated as in Algorithm 2 with*

$$
k = \max\left\{ O\left(\frac{\log^2 n}{\epsilon^2} \right), \left(\frac{1}{\epsilon} \right)^{O(1/\epsilon^2)} \right\}.
$$

Then, we have that with probability at least $2/3$, the following hold: Let $\tilde{\boldsymbol{v}}^*$ and $\tilde{z}^*$ be an optimal solution and the optimal value, respectively, of the problem

$$
\min_{\|v\|_2 \leq \sqrt{\frac{|S|}{n}} r} \psi_{|S|, A|_S, d|_S, b|_S}(\boldsymbol{v}) \,.
$$

Then,

$$
\left| \frac{1}{|S|^2} \tilde{z}^* - \frac{1}{n^2} z^* \right| \leq \frac{\epsilon L r^2}{n} \,,
$$

where $L = \max\{\max_{i,j} |A_{ij}|, \max_i |d_i|, \max_i |b_i|\}$.

References

1. Farid Alizadeh. Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM journal on Optimization*, 5(1):13–51, 1995.
2. Aharon Ben-Tal and Marc Teboulle. Hidden convexity in some nonconvex quadratically constrained quadratic programming. *Mathematical Programming*, 72(1):51–63, 1996.
3. Léon Bottou. Stochastic learning. In *Advanced Lectures on Machine Learning*, pages 146–168. Springer, 2004.
4. Stanislav Busygin. A new trust region technique for the maximum weight clique problem. *Discrete Applied Mathematics*, 154(15):2080–2096, 2006.
5. Kenneth L. Clarkson, Elad Hazan, and David P. Woodruff. Sublinear optimization for machine learning. *Journal of the ACM*, 59(5):23:1–23:49, 2012.
6. A Frieze and R Kannan. The regularity lemma and approximation schemes for dense problems. In *Proceedings of the 37th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 12–20, 1996.
7. Alan Frieze and Ravi Kannan. Quick approximation to matrices and applications. *Combinatorica*, 19(2):175–220, 1999.
8. Alan Frieze, Ravi Kannan, and Santosh Vempala. Fast monte-carlo algorithms for finding low-rank approximations. *Journal of the ACM (JACM)*, 51(6):1025–1041, 2004.
9. Walter Gander, Gene H Golub, and Urs von Matt. A constrained eigenvalue problem. *Linear Algebra and its applications*, 114:815–839, 1989.
10. Kohei Hayashi and Yuichi Yoshida. Minimizing quadratic functions in constant time. In *Proceedings of the 30th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 2217–2225, 2016.
11. Kohei Hayashi and Yuichi Yoshida. Fitting low-rank tensors in constant time. In *Proceedings of the 31th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 2473–2481, 2017.
12. Elad Hazan and Tomer Koren. A linear-time algorithm for trust region problems. *Mathematical Programming*, 158(1-2):363–381, 2016.
13. L Lovász. *Large Networks and Graph Limits*. American Mathematical Society, 2012.
14. László Lovász and Balázs Szegedy. Limits of dense graph sequences. *Journal of Combinatorial Theory, Series B*, 96(6):933–957, 2006.
15. David G. Luenberger. *Optimization by Vector Space Methods*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1997.
16. Claire Mathieu and Warren Schudy. Yet another algorithm for dense max cut: go greedy. In *Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 176–182, 2008.
17. Kevin P Murphy. *Machine learning: a probabilistic perspective*. The MIT Press, 2012.
18. Yurii Nesterov. A method of solving a convex programming problem with convergence rate o (1/k2). In *Soviet Mathematics Doklady*, volume 27, pages 372–376, 1983.
19. Yurii Nesterov and Arkadii Nemirovskii. *Interior-point polynomial algorithms in convex programming*. SIAM, 1994.
20. Vladimir Nikiforov. Cut-norms and spectra of matrices. *arXiv:0912.0336*, 2009.
21. Anthony L. Peressini, Francis E. Sullivan, and J. J. Jr. Uhl. *The Mathematics of Nonlinear Programming*. Springer, 1993.
22. Joel A Tropp. An introduction to matrix concentration inequalities. *Foundations and Trends ® in Machine Learning*, 8:1–230, 2015.
23. Yinyu Ye and Shuzhong Zhang. New results on quadratic minimization. *SIAM Journal on Optimization*, 14(1):245–267, 2003.
24. Hongchao Zhang, Andrew R Conn, and Katya Scheinberg. A derivative-free algorithm for least-squares minimization. *SIAM Journal on Optimization*, 20(6):3555–3576, 2010.

A Omitted Technical Lemmas and Proofs

A.1 Dikernels and Sampling Lemmas

Proof of Lemma 12: Before starting the proof we introduce some notations and an important observation. We note that $\sup_{f:[0,1]\to\mathbb{R}} \frac{\|\mathcal{A}f\|_2^2}{\|f\|_2^2}$ has a minimizer, since the objective function is weakly continuous and we may assume that $\|f\|_2 \leq 1$ which is weakly compact.

For every $x \in [0,1]$ and $j \in [m]$, we let $\mathcal{A}(x, I_j^m) = A_{i^n(x)j}$. Also for every $(i,j) \in [n]\times[m]$, we let $\mathcal{A}(I_i^n, I_j^m) = A_{ij}$.

We start by showing that $\max_{\boldsymbol{v}\in\mathbb{R}^m} \frac{\|A\boldsymbol{v}\|_2^2}{\|\boldsymbol{v}\|_2^2} \leq nm \cdot \sup_{f:[0,1]\to\mathbb{R}} \frac{\|\mathcal{A}f\|_2^2}{\|f\|_2^2}$. Given a vector $\boldsymbol{v} \in \mathbb{R}^m$, we define the function $f : [0,1] \to \mathbb{R}$ as $f(x) = v_{i^m(x)}$. Then,

$$\begin{aligned}
\|\mathcal{A}f\|_2^2 &= \int_0^1 \left(\int_0^1 \mathcal{A}(x,y)f(y)dy\right)^2 dx = \int_0^1 \left(\sum_{j\in[m]} \int_{I_j^m} \mathcal{A}(x,y)f(y)dy\right)^2 dx \\
&= \int_0^1 \left(\frac{1}{m}\sum_{j\in[m]} \mathcal{A}(x, I_j^m)v_j\right)^2 dx = \frac{1}{n}\sum_{i\in[n]} \left(\frac{1}{m}\sum_{j\in[m]} \mathcal{A}(I_i^n, I_j^m)v_j\right)^2 \\
&= \frac{1}{n}\sum_{i\in[n]} \left(\frac{1}{m}\sum_{j\in[m]} A_{ij}v_j\right)^2 = \frac{1}{nm^2}\|A\boldsymbol{v}\|_2^2 \,.
\end{aligned}$$

In addition,

$$\|f\|_2^2 = \int_0^1 f(x)^2 dx = \sum_{j\in[m]} \int_{I_j^m} f(x)^2 dx = \frac{1}{m}\sum_{j\in[m]} v_j^2 = \frac{1}{m}\|\boldsymbol{v}\|_2^2 \,.$$

Next, we will prove that $\max_{\boldsymbol{v}\in\mathbb{R}^m} \frac{\|A\boldsymbol{v}\|_2^2}{\|\boldsymbol{v}\|_2^2} \geq nm \cdot \sup_{f:[0,1]\to\mathbb{R}} \frac{\|\mathcal{A}f\|_2^2}{\|f\|_2^2}$. Let $f : [0,1] \to \mathbb{R}$ be a measurable function. Then, for any $x \in [0,1]$, consider the partial derivative,

$$\frac{\partial}{\partial f(x)} \frac{\|\mathcal{A}f\|_2^2}{\|f\|_2^2} = \frac{\|f\|_2^2 \cdot \frac{\partial}{\partial f(x)}\|\mathcal{A}f\|_2^2 - \|\mathcal{A}f\|_2^2 \cdot \frac{\partial}{\partial f(x)}\|f\|_2^2}{\|f\|_2^4} \,.$$

Note that,

$$\frac{\partial}{\partial f(x)}\|\mathcal{A}f\|_2^2 = \frac{\partial}{\partial f(x)}\langle \mathcal{A}f, \mathcal{A}f\rangle = \frac{\partial}{\partial f(x)}\langle f, (\mathcal{A}^*\mathcal{A})f\rangle \,,$$

where $\mathcal{A}^*(x,y) = \mathcal{A}^*(y,x)$. So, for $\mathcal{M} = \mathcal{A}^*\mathcal{A}$, we have

$$\begin{aligned}
\frac{\partial}{\partial f(x)}\|\mathcal{A}f\|_2^2 &= \frac{\partial}{\partial f(x)}\langle f, \mathcal{M}f\rangle \\
&= \sum_{j\in[m]} \int_{I_j^m} \mathcal{M}(I_j^m, i^m(x))f(y)dy + \sum_{j\in[m]} \int_{I_j^m} \mathcal{M}(i^m(x), I_j^m)f(y)dy \,.
\end{aligned}$$

Therefore,

$$\frac{\partial}{\partial f(x)}\frac{\|\mathcal{A}f\|_2^2}{\|f\|_2^2} = \frac{\|f\|_2^2\left(\sum_{j\in[m]}\int_{I_j^m}\mathcal{M}(I_j^m,i^m(x))f(y)dy+\sum_{j\in[m]}\int_{I_j^m}\mathcal{M}(i^n(x),I_j^m)f(y)dy\right)}{\|f\|_2^4} - \frac{2\|\mathcal{A}f\|_2^2\cdot f(x)}{\|f\|_2^4}\,.$$

Consider the optimal solution f^*. By the form of the partial derivative, it holds that $f^*(z)=f^*(z')$ for almost all $z,z'\in[0,1]$ such that $i_n(z)=i_n(z')$. That is, f^* is almost constant on each of the intervals $I_1^m,\ldots,I_m^m$. Hence, we can define $\boldsymbol{v}\in\mathbb{R}^m$ as $v_j=f^*(x)$, where x is the dominant value in I_j^m. Then,

$$\|A\boldsymbol{v}\|_2^2=\sum_{i\in[n]}\left(\sum_{j\in[m]}A_{ij}v_j\right)^2=nm^2\int_0^1\left(\sum_{j\in[m]}\int_{I_j^m}\mathcal{A}(x,I_j^m)f^*(y)dy\right)^2dx$$
$$=nm^2\int_0^1\left(\int_0^1\mathcal{A}(x,y)f^*(y)dy\right)^2dx=nm^2\|\mathcal{A}f^*\|_2^2$$

Moreover,

$$\|f^*\|_2^2=\int_0^1 f^N(x)^2dx=\sum_{j\in[m]}\int_{I_j^m}f^*(x)^2dx=\frac{1}{m}\sum_{j\in[m]}v_j^2$$

Therefore, we have $\max_{v\in\mathbb{R}^m}\frac{\|A\boldsymbol{v}\|_2^2}{\|\boldsymbol{v}\|_2^2}\geq nm\cdot\sup_{f:[0,1]\to\mathbb{R}}\frac{\|\mathcal{A}f\|_2^2}{\|f\|_2^2}$. ◀

Proof of Lemma 15: We first note that for any $t\in[T]$, any refinement of the row or column partitions of A_t^{str} will give the same block approximation matrix A_t^{str}. Therefore, let $\mathcal{P}_R=\{V_1^R,\ldots,V_P^R\}$ be a partition which refines $\mathcal{P}_R^{A_1},\ldots,\mathcal{P}_R^{A_T}$ and its size is $P=O(p^T)$. Similarly, let $\mathcal{P}_C=\{V_1^C,\ldots,V_Q^C\}$ be a partition which refines $\mathcal{P}_C^{A_1},\ldots,\mathcal{P}_C^{A_T}$ and its size is $Q=O(q^T)$.

We denote by z_i^R the number of elements of S_R that falls into the set V_i^R. Then,

$$\mathop{\mathbf{E}}_{S_R}[z_i^R]=s_R\cdot\mu(V_i^R)\ \text{ and }\ \mathbf{Var}[z_i^R]=\mu(V_i^R)(1-\mu(V_i^R))s_R\,.$$

Similarly, we denote by z_j^C the number of elements of S_C that falls into the set V_j^C. Then,

$$\mathop{\mathbf{E}}_{S_C}[z_j^C]=s_C\cdot\mu(V_j^C)\ \text{ and }\ \mathbf{Var}[z_j^C]=\mu(V_j^C)(1-\mu(V_j^C))s_C\,.$$

We next construct a measure preserving bijection. We define the following two partitions of the $[0,1]$ interval.

Let $\{V_1^{R'},\ldots,V_P^{R'}\}$ be a partition such that $\mu(V_i^{R'})=z_i^R/s_R$ and $\mu(V_i^R\cap V_i^{R'})=\min(\mu(V_i^R),z_i^R/s_R)$, and let $\{V_1^{C'},\ldots,V_Q^{C'}\}$ be a partition such that $\mu(V_j^{C'})=z_j^C/s_C$ and $\mu(V_j^C\cap V_j^{C'})=\min(\mu(V_j^C),z_j^C/s_C)$. We construct the dikernel $\mathcal{Y}:[0,1]^2\to\mathbb{R}$ such that the value of $\mathcal{Y}$ on $V_i^{R'}\times V_j^{C'}$ is the same as the value of $\mathcal{A}^{\mathrm{str}}$ on $V_i^R\times V_j^C$. Therefore, the dikernel $\mathcal{Y}$ agrees with $\mathcal{A}^{\mathrm{str}}$ on the set $Y=\bigcup_{(i,j)\in[P]\times[Q]}(V_i^R\cap V_i^{R'})\times(V_j^C\cap V_j^{C'})$. Then,

there exists a bijection π such that $\pi(\mathcal{A}^{\text{str}}|_{S_R \times S_C}) = \mathcal{Y}$. Then,

$$\begin{aligned}
1 - \mu(Y) &\leq 1 - \left(\sum_{i \in [P]} \min\left(\mu(V_i^R), \frac{z_i^R}{s_R} \right) \right) \left(\sum_{j \in [Q]} \min\left(\mu(V_j^C), \frac{z_j^C}{s_C} \right) \right) \\
&= 1 - \left(1 - \frac{1}{2} \sum_{i \in [P]} \left| \mu(V_i^R) - \frac{z_i^R}{s_R} \right| \right) \left(1 - \frac{1}{2} \sum_{j \in [Q]} \left| \mu(V_j^C) - \frac{z_j^C}{s_C} \right| \right) \\
&\leq \max \left\{ \left(P \sum_{i \in [P]} \left(\mu(V_i^R) - \frac{z_i^R}{s_R} \right)^2 \right)^{1/2}, \left(Q \sum_{j \in [Q]} \left(\mu(V_j^C) - \frac{z_j^C}{s_C} \right)^2 \right)^{1/2} \right\} .
\end{aligned}$$

Therefore, we have that

$$(1 - \mu(Y))^2 \leq \max \left\{ P \sum_{i \in [P]} \left(\mu(V_i^R) - \frac{z_i^R}{s_R} \right)^2, Q \sum_{j \in [Q]} \left(\mu(V_j^C) - \frac{z_j^C}{s_C} \right)^2 \right\} .$$

Taking expectation (over the choice of S_R and S_C) yields,

$$\mathop{\mathbf{E}}_{S_R, S_C} [1 - \mu(Y)] \leq \max \left(\sqrt{\frac{Q}{s_C}}, \sqrt{\frac{P}{s_R}} \right) .$$

Let $\mathcal{U} = \mathcal{A}^{\text{str}} - \mathcal{Y}$ and consider a corresponding matrix U. U is an $N \times M$ matrix, where $N = \text{lcm}(n \cdot \mu(V_1^R \triangle V_1^{R'}), \ldots, n \cdot \mu(V_P^R \triangle V_P^{R'}))$ and $M = \text{lcm}(m \cdot \mu(V_1^C \triangle V_1^{C'}), \ldots, m \cdot \mu(V_Q^C \triangle V_Q^{C'}))$. By Claim 10, the absolute value of an entry in the matrix A^{str} is bounded by $\frac{2}{\gamma^{11}} L$, and thus the absolute value of an entry in U is bounded by $\frac{4}{\gamma^{11}} L$.

Then,

$$\begin{aligned}
\mathop{\mathbf{E}}_{S_R, S_C} [\|U\|_2^2] &\leq \mathop{\mathbf{E}}_{S_R, S_C} [\|U\|_F^2] = \mathop{\mathbf{E}}_{S_R, S_C} \left[\sum_{i=1}^{N} \sum_{j=1}^{M} U_{ij}^2 \right] \\
&\leq \mathop{\mathbf{E}}_{S_R, S_C} \left[NM(1 - \mu(Y)) \cdot \max_{(i,j) \in [N] \times [M]} (U)_{ij}^2 \right] \\
&\leq NM \cdot \max_{(i,j) \in [N] \times [M]} U_{ij}^2 \cdot \mathop{\mathbf{E}}_{S_R, S_C} [1 - \mu(Y)] \\
&\leq NM \cdot \left(\frac{4}{\gamma^{11}} \right)^2 L^2 \cdot \max \left(\sqrt{\frac{Q}{s_C}}, \sqrt{\frac{P}{s_R}} \right) .
\end{aligned}$$

Using Corollary 13 we get $\mathop{\mathbf{E}}_{S_R, S_C} [\|\mathcal{U}\|_2] \leq \frac{4L}{\gamma^{11}} \max \left(\sqrt{\frac{p^{T/2}}{s_R^{1/2}}}, \sqrt{\frac{q^{T/2}}{s_C^{1/2}}} \right)$, and the lemma follows. ◀

In the remaining part of the subsection we will prove Lemma 16. In order to prove the lemma we introduce the following result regarding a random submatrix from [22] section 5.2.2.

▶ **Lemma 19.** *Given a matrix $A \in [-L, L]^{n \times m}$, let $P = \text{diag}(\chi_1, \ldots, \chi_n)$ be the diagonal matrix where $\{\chi_i\}$'s are Bernoulli(k_R/n) random variables for $k_R > 0$. In addition, let $R = \text{diag}(\xi_1, \ldots, \xi_m)$ be the diagonal matrix where $\{\xi_j\}$'s are Bernoulli(k_C/m) random variables for $k_C > 0$. Then,*

$$\mathbf{E}[\|PAR\|_2^2] \leq \frac{3k_R \cdot k_C}{nm} \|A\|_2^2 + 2k_R L^2 \log n + 2k_C L^2 \log m + L^2 \log n \log m .$$

The above lemma shows that a random submatrix of size roughly $k_R \times k_C$ gets its "fair share" of the spectral norm of A.

Proof of Lemma 16: Since S_R and S_C are set of indices generated by choosing each index to S_R (or S_C) with probability k_R/n (k_C/m), we have that with probability at least 99/100,

$$|S_R| \geq k_R/2 \text{ and } |S_C| \geq k_C/2 \,.$$

We henceforth condition on that.

For any measure preserving bijection $\pi : [0,1] \to [0,1]$, and $t \in [T]$ we have

$$\begin{aligned}\mathop{\mathbf{E}}_{S_R,S_C}[\|\mathcal{A}_t - \pi(\mathcal{A}_t|_{S_R \times S_C})\|_2] \leq \|\mathcal{A}_t - \mathcal{A}_t^{\mathrm{str}}\|_2 + \mathop{\mathbf{E}}_{S_R,S_C}\left[\|\mathcal{A}_t^{\mathrm{str}} - \pi(\mathcal{A}_t^{\mathrm{str}}|_{S_R \times S_C})\|_2\right] \\ + \mathop{\mathbf{E}}_{S_R,S_C}\left[\|\pi(\mathcal{A}_t^{\mathrm{str}}|_{S_R \times S_C}) - \pi(\mathcal{A}_t|_{S_R \times S_C})\|_2\right] \,.\end{aligned}$$

Where A_t^{str} is the matrix obtained by Lemma 8.

By Lemma 8 and Corollary 13, we have that for any $t \in [T]$

$$\|\mathcal{A}_t - \mathcal{A}_t^{\mathrm{str}}\|_2 \leq 7\gamma L \,.$$

By the facts that $|S_R| \geq k_R/2$, $|S_C| \geq k_C/2$, $p = O\left(\left(\frac{1}{\gamma^{10}}\right)^{3/\gamma^2}\right)$ and $q = O\left(\left(\frac{1}{\gamma^{10}}\right)^{3/\gamma^2}\right)$, Lemma 15 yields that for any $t \in [T]$:

$$\mathop{\mathbf{E}}_{S_R,S_C}\left[\|\mathcal{A}_t^{\mathrm{str}} - \pi(\mathcal{A}_t^{\mathrm{str}}|_{S_R \times S_C})\|_2\right] = O\left(\frac{L}{\gamma^{11}}\left(\frac{1}{\gamma^{10}}\right)^{\frac{3T}{4\gamma^2}} \cdot \max\left(\frac{1}{k_R^{1/4}}, \frac{1}{k_C^{1/4}}\right)\right)$$

We are left with bounding $\mathop{\mathbf{E}}_{S_R,S_C}[\|\pi(\mathcal{A}_t^{\mathrm{str}}|_{S_R \times S_C}) - \pi(\mathcal{A}_t|_{S_R \times S_C})\|_2]$. We apply Lemma 19 on $(A_t^{\mathrm{str}} - A_t)|_{S_R \times S_C}$ to get

$$\begin{aligned}&\mathop{\mathbf{E}}_{S_R,S_C}\left[\|(A_t^{\mathrm{str}} - A_t)|_{S_R \times S_C}\|_2^2\right] \\ &\quad\leq \frac{3k_C \cdot k_R}{nm}\|A^{\mathrm{str}} - A\|_2^2 + 2L^2 k_R \log n + 2L^2 k_C \log m + L^2 \log n \log m \\ &\quad\leq \frac{3k_C \cdot k_R}{nm} \cdot 16\gamma^2 L^2 nm + 2L^2 k_R \log n + 2L^2 k_C \log m + L^2 \log n \log m \\ &\quad\leq 48L^2\gamma^2 k_R \cdot k_C + 2L^2 k_R \log n + 2L^2 k_C \log m + L^2 \log n \log m \,,\end{aligned}$$

which implies that,

$$\begin{aligned}&\mathop{\mathbf{E}}_{S_R,S_C}\left[\|(A_t^{\mathrm{str}} - A_t)|_{S_R \times S_C}\|_2\right] \\ &\quad\leq 7\gamma L\sqrt{k_C \cdot k_R} + L\sqrt{2k_R \log n} + L\sqrt{2k_C \log m} + L\sqrt{\log m \log n} \,.\end{aligned}$$

By applying Corollary 13 and using the fact that the dimension of $(A^{\mathrm{str}} - A)|_{S_R \times S_C}$ at least $k_R \cdot k_C/4$, we get

$$\mathop{\mathbf{E}}_{S_R,S_C}\left[\|\mathcal{A}_t^{\mathrm{str}}|_{S_R \times S_C} - \mathcal{A}_t|_{S_R \times S_C}\|_2\right] \leq 14\gamma L + L\sqrt{\frac{8\log n}{k_c}} + L\sqrt{\frac{8\log m}{k_R}} + L\sqrt{\frac{4\log m \log n}{k_R \cdot k_C}} \,.$$

Putting everything together,

$$\begin{aligned}\mathop{\mathbf{E}}_{S_R,S_C}[\|\mathcal{A}_t - \pi(\mathcal{A}_t|_{S_R \times S_C})\|_2] \leq 21\gamma L + L\left(\sqrt{\frac{8\log n}{k_C}} + \sqrt{\frac{8\log m}{k_R}} + \sqrt{\frac{4\log n \log m}{k_R k_C}}\right) \\ + O\left(\frac{L}{\gamma^{11}}\left(\frac{1}{\gamma^{10}}\right)^{\frac{3T}{4\gamma^2}} \cdot \max\left(\frac{1}{k_R^{1/4}}, \frac{1}{k_C^{1/4}}\right)\right) \,.\end{aligned}$$

By Markov inequality, with probability at least $9/10T$,

$$\|\mathcal{A}_t - \pi(\mathcal{A}_t|_{S_R \times S_C})\|_2 \leq 210\gamma LT + 10LT\left(\sqrt{\frac{8\log n}{k_C}} + \sqrt{\frac{8\log m}{k_R}} + \sqrt{\frac{4\log n \log m}{k_R k_C}}\right) + O\left(\frac{LT}{\gamma^{11}}\left(\frac{1}{\gamma^{10}}\right)^{\frac{3T}{4\gamma^2}} \cdot \max\left(\frac{1}{k_R^{1/4}}, \frac{1}{k_C^{1/4}}\right)\right) .$$

By using a union bound the lemma follows. ◀

A.2 Quadratic Function Minimization

Proof of Lemma 17: In contrast to the proof of Lemma 12, in this case we have to deal with constrained optimization, and therefore must consider the KKT optimality conditions. We start by showing that $\min_{\boldsymbol{v}:\|\boldsymbol{v}\|_2 \leq r} \psi_{n,A,\boldsymbol{d},\boldsymbol{b}}(\boldsymbol{v}) \geq n^2 \cdot \inf_{f:\|f\|_2 \leq \frac{r}{\sqrt{n}}} \Psi_{n,A,\boldsymbol{d},\boldsymbol{b}}(f)$. Given a vector $\boldsymbol{v} \in \mathbb{R}^n$ such that $\|\boldsymbol{v}\|_2 \leq r$, we define the function $f : [0,1] \to \mathbb{R}$ as $f(x) = v_{i^n(x)}$. Then,

$$\langle f, \mathcal{A}f\rangle = \sum_{i,j\in[n]} \int_{I_i^n}\int_{I_j^n} A_{ij} f(x) f(y) dx dy = \frac{1}{n^2}\langle \boldsymbol{v}, A\boldsymbol{v}\rangle$$
$$\left\langle f^2, \mathcal{D}\mathbf{1}\right\rangle = \sum_{i,j\in[n]} \int_{I_i^n}\int_{I_j^n} d_i f(x)^2 dx dy = \frac{1}{n}\langle \boldsymbol{v}, \mathrm{diag}(\boldsymbol{d})\boldsymbol{v}\rangle$$
$$\langle f, \mathcal{B}\mathbf{1}\rangle = \sum_{i,j\in[n]} \int_{I_i^n}\int_{I_j^n} b_i f(x) dx dy = \frac{1}{n}\langle \boldsymbol{v}, \boldsymbol{b}\rangle$$

In addition,

$$\|f\|_2^2 = \int_0^1 f(x)^2 dx = \sum_{j\in[n]} \int_{I_j^n} f(x)^2 dx = \frac{1}{n}\sum_{j\in[n]} v_j^2 = \frac{1}{n}\|\boldsymbol{v}\|_2^2 \leq \frac{r^2}{n} .$$

Next, we show that $\min_{\boldsymbol{v}:\|\boldsymbol{v}\|_2 \leq r} \psi_{n,A,\boldsymbol{d},\boldsymbol{b}}(\boldsymbol{v}) \leq n^2 \cdot \inf_{f:\|f\|_2 \leq \frac{r}{\sqrt{n}}} \Psi_{n,A,\boldsymbol{d},\boldsymbol{b}}(f)$. First, we note that the latter problem has a minimizer $f : [0,1] \to \mathbb{R}$ because it is weakly continuous and coercive (See, e.g., [21]). According to the generalized KKT conditions (see, e.g., Section 9.4 of [15]), there exists λ such that:

- (Stationarity) $\frac{\partial}{\partial f^*(x)}\Psi_{n,A,\boldsymbol{d},\boldsymbol{b}}(f^*(x)) - \lambda\frac{\partial}{\partial f^*(x)}(\|f^*\|_2 - r/\sqrt{n}) = 0$ for almost all x.
- (Primal feasibility) $\|f^*\|_2 - r/\sqrt{n} \leq 0$
- (Complementary slackness) $\lambda \cdot (\|f^*\|_2 - r/\sqrt{n}) = 0$

The stationarity condition yields:

$$\frac{\partial}{\partial f^*(x)}\Psi_{n,A,\boldsymbol{d},\boldsymbol{b}}(f^*(x)) - \lambda\frac{\partial}{\partial f^*(x)}(\|f^*\|_2 - r/\sqrt{n})$$
$$= \sum_{i\in n}\int_{I_i^n} A_{i i^n(x)} f^*(y) dy + \sum_{j\in n}\int_{I_j^n} A_{i^n(x) j} f^*(y) dy + 2d_{i^n(x)} f^*(x) + b_{i^n(x)} - 2\lambda f^*(x) ,$$

By the form of the partial derivatives, $f^*(z) = f^*(z')$ for almost all $z, z' \in [0,1]$ such that $i^n(z) = i^n(z')$. That is, f^* is almost constant on each of the intervals $I_1^n, \ldots, I_n^n$. Therefore,

we define $\boldsymbol{v} \in \mathbb{R}^n$ as $v_j = f^*(x)$, where x is the dominant value in I_j^n. Then,

$$\langle \boldsymbol{v}, A\boldsymbol{v} \rangle = \sum_{i,j \in [n]} A_{ij} v_i v_j = n^2 \sum_{i,j \in [n]} \int_{I_i^n} \int_{I_j^n} A_{ij} f^*(x) f^*(y) dx dy = n^2 \langle f^*, \mathcal{A} f^* \rangle \ .$$

$$\langle \boldsymbol{v}, \mathrm{diag}(\boldsymbol{d}) \boldsymbol{v} \rangle = \sum_{i \in [n]} d_i v_i^2 = n \sum_{i \in [n]} \int_{I_i^n} d_i f^*(x)^2 dx = n \left\langle (f^*)^2, \mathcal{D}\mathbf{1} \right\rangle \ .$$

$$\langle \boldsymbol{v}, \boldsymbol{b} \rangle = \sum_{i \in [n]} b_i v_i = n \sum_{i \in [n]} \int_{I_i^n} b_i f^*(x) dx = n \langle f^*, \mathcal{B}\mathbf{1} \rangle \, .$$

In addition, $\|f^*\|_2^2 = \int_0^1 f^*(x)^2 dx = \sum_{j \in [n]} \int_{I_j^n} f^*(x)^2 dx = \frac{1}{n} \|\boldsymbol{v}\|_2^2 \leq \frac{r^2}{n}$.
Hence, we get that

$$\min_{\boldsymbol{v}: \|\boldsymbol{v}\|_2 \leq r} \psi_{n,A,\boldsymbol{d},\boldsymbol{b}}(\boldsymbol{v}) \leq n^2 \cdot \inf_{f: \|f\|_2 \leq \frac{r}{\sqrt{n}}} \Psi_{n,A,\boldsymbol{d},\boldsymbol{b}}(f) \ ,$$

and the lemma follows. ◀

Deterministic Heavy Hitters with Sublinear Query Time

Yi Li
Nanyang Technological University, Singapore
yili@ntu.edu.sg

Vasileios Nakos[1]
Harvard University, USA
vasileiosnakos@g.harvard.edu

Abstract

We study the classic problem of finding ℓ_1 heavy hitters in the streaming model. In the general turnstile model, we give the first deterministic sublinear-time sketching algorithm which takes a linear sketch of length $\mathcal{O}(\epsilon^{-2}\log n\cdot\log^*(\epsilon^{-1}))$, which is only a factor of $\log^*(\epsilon^{-1})$ more than the best existing polynomial-time sketching algorithm (Nelson et al., RANDOM '12). Our approach is based on an iterative procedure, where most unrecovered heavy hitters are identified in each iteration. Although this technique has been extensively employed in the related problem of sparse recovery, this is the first time, to the best of our knowledge, that it has been used in the context of heavy hitters. Along the way we also obtain a sublinear time algorithm for the closely related problem of the ℓ_1/ℓ_1 compressed sensing, matching the space usage of previous (super-)linear time algorithms. In the strict turnstile model, we show that the runtime can be improved and the sketching matrix can be made strongly explicit with $O(\epsilon^{-2}\log^3 n/\log^3(1/\epsilon))$ rows.

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

Keywords and phrases heavy hitters, turnstile model, sketching algorithm, strongly explicit

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.18

Related Version Full version available at `https://arxiv.org/abs/1712.01971`.

1 Introduction

The problem of detecting *heavy hitters*, also frequently referred to as *elephants* or *hot items*, is one of the most well-studied problems in databases and data streams, from both theoretical and practical perspectives. In this problem, we are given a long data stream of elements coming from a large universe, and we are asked to report all the elements that appear at least a large number of times (called *heavy hitters*), using space that is much smaller than the size of the universe and the length of the stream.

Finding popular terms in search queries, identifying destination adresses of packets, detecting anomalies in network traffic streams such as denial-of-service (DoS) attacks, or performing traffic engineering, are only some of the important practical appearances of the heavy hitters problem. For example, the central task of managing large-scale networks lies in accurately measuring and monitoring network traffic [28, 26]. Interestingly, empirical studies [6, 21, 23, 27] indicate that flow-statistics in large networks follow an elephant/mice

[1] Supported in part by NSF grant IIS-1447471

phenomenon, i.e., the vast majority of the bytes are concentrated on only a small fraction of the flows.

On the theoretical side, heavy hitters appear very often, both in streaming algorithms and sparse recovery tasks. For the problems such as streaming entropy estimation [11], ℓ_p sampling [1, 20], finding duplicates [14], block heavy hitters [13], sparse recovery tasks [8, 9, 10], many algorithmic solutions use heavy hitters algorithms as subroutines.

Streaming Models. In this paper we consider the most general streaming model, called the *(general) turnstile model*, defined as follows. There is an underlying vector $x \in \mathbb{R}^n$, which is initialized to zero and is maintained throughout the input stream. Each element in the input stream describes an update $x_i \leftarrow x_i + \delta$ for some index i and increment δ, where δ can be either positive or negative.

We also consider a restricted version of the general turnstile model, called the *strict turnstile model*, under which it is guaranteed that $x_i \geq 0$ for all i throughout the input stream. This restricted model captures the practical scenario where item deletions are allowed but an item cannot be deleted more than it is inserted.

Sketching Algorithms. An important class of streaming algorithms is called *sketching algorithms*. A sketching algorithm maintains a short linear sketch $v = \Phi x$ (where $\Phi \in \mathbb{R}^{m \times n}$) throughout the input stream and then runs a recovery algorithm $\mathcal{D}$, which has access to only v and Φ, to output a desired $\hat{x}$. The space usage is proportional to m (the length of the sketch v) and to the memory needed to store Φ. Therefore we wish to minimize m and design a structured Φ such that storing Φ takes little space. Surprisingly, all existing streaming algorithms under the general turnstile model are sketching algorithms, and it has been shown [17] that all streaming algorithms under the general turnstile model can be converted to sketching algorithms with a mild increase in the space usage.

1.1 ℓ_∞/ℓ_1 Sparse Recovery

The heavy hitter problem has been studied under various streaming models and various recovery guarantees. Depending on the heaviness we are interested in, we distinguish between ℓ_1 and ℓ_2 heavy hitters. We are interested in finding, in the first case, the coordinates which are at least $\epsilon\|x\|_1$ in magnitude, and in the second case, the coordinates which are at least $\epsilon\|x\|_2$. Although finding ℓ_2 heavy hitters is strictly stronger than finding ℓ_1 heavy hitters, we consider only ℓ_1 heavy hitters in this paper, for it is impossible to find ℓ_2 heavy hitters using a deterministic space-saving sketching algorithm (see details below).

Specifically, we consider a classical recovery guarantee, called the ℓ_∞/ℓ_1 error guarantee in the literature, that is, the algorithm outputs an $\mathcal{O}(1/\epsilon)$-sparse vector $\hat{x}$ such that

$$\|\hat{x} - x\|_\infty \leq \epsilon\|x_{-r}\|_1, \tag{1}$$

for some parameters ϵ and r, where x_{-r} is the vector obtained by zeroing out the largest r coordinates of x in magnitude (absolute value). This type of guarantee requires not only finding the heavy hitters, but also giving 'good enough' estimates of them, where the estimates are measured with respect to $\|x_{-r}\|_1$ instead of the larger $\|x\|_1$; this type of guarantee is called the *tail guarantee*. It should be noted that the ℓ_∞/ℓ_1 guarantee has been extensively studied and is provided by several classical algorithms, e.g. Count-Min [5], LossyCounting [18], SpaceSaving [19], although not all of them work under the general turnstile model.

In this paper we focus on deterministic sketching algorithms, which means that both the matrix Φ and the recovery algorithm $\mathcal{D}$ allow uniform reconstruction of every $x \in \mathbb{R}^n$ up to $\epsilon\|x_{-r}\|_1$ error, providing the best applicability. This is also referred to as "for-all" guarantee in the sparse recovery literature, in contrast to the "for-each" guarantee, which allows reconstruction of a fixed vector with some target success probability. Most previous sketching algorithms for the heavy hitter problems concern the "for-each" model and resort to randomization (e.g. [5, 3]), by drawing a random Φ from some distribution and guaranteeing $\mathcal{D}$ to output an acceptable $\hat{x}$ with a good probability. Other sketching algorithms are deterministic, however, they run in time at least linear in the universe size n. The goal of fast query time, say, logarithmic in n, is crucial to streaming applications. For instance, in traffic monitoring n equals the number of all possible packets, namely 2^{32}; a linear runtime would be prohibitive in any reasonable real-world scenario. A natural goal is to design a sketching algorithm with sublinear query time, preferably $\mathcal{O}(\text{poly}(1/\epsilon, r, \log n))$, with as little space usage as possible.

Apart from practical importance, deterministic algorithms for heavy hitters is an interesting theoretical subfield of streaming algorithms, connected to dimensionality reduction and incoherent matrices [22]. Moreover, gaining insight into such questions may give insight for many other data stream problems where heavy hitter algorithms are used as subroutines. We note that any deterministic sketching algorithm that finds ℓ_2 heavy hitters requires $\Omega(n)$ space [4] (which implies that the trivial algorithm storing the entire input vector x is asymptotically optimal), while the best lower bound for ℓ_1 heavy hitters is $\Omega(r\log(n/r)/\log r + \epsilon^{-2} + \epsilon^{-1}\log n)$ [22, 7].

The state-of-the-art deterministic sketching algorithms for ℓ_1 heavy hitters are found in [22], where two algorithms are given. The first algorithm uses $m = \mathcal{O}(\epsilon^{-2}\log n \cdot \min\{1, \log n/(\log\log n + \log(1/\epsilon))^2\})$ rows and achieves $\epsilon\|x_{-\lceil 1/\epsilon\rceil}\|_1$ tail guarantee (setting $r = \lceil 1/\epsilon\rceil$ in (1)). The second algorithm uses $m = \mathcal{O}(\epsilon^{-2}\log n)$ rows and achieves a stronger $\epsilon\|x_{-\lceil 1/\epsilon^2\rceil}\|_1$ tail guarantee. We can see that when $\epsilon < 2^{-\Omega(\sqrt{\log n})}$, the first algorithm uses less space, whereas when $\epsilon \geq 2^{-\Omega(\sqrt{\log n})}$ the second algorithm is better. The number of rows used by both algorithms is at most suboptimal by a $\log n$ factor while the runtimes are both superlinear $\Omega(\epsilon^{-1}n\log n)$. In this paper our goal is to obtain an algorithm which runs in sublinear time in n while attaining the stronger $\epsilon\|x_{-\lceil 1/\epsilon^2\rceil}\|_1$ tail guarantee with near-optimal number of rows. Our main theorem is formally stated below.

▶ **Theorem 1** (ℓ_∞/ℓ_1). *There exists a linear sketch $\Phi \in \mathbb{R}^{m\times n}$ such that for every $x \in \mathbb{R}^n$, we can, given Φx, find an $\mathcal{O}(1/\epsilon)$-sparse vector $\hat{x}$ such that*

$$\|x - \hat{x}\|_\infty \leq \epsilon\|x_{-\lceil 1/\epsilon^2\rceil}\|_1,$$

in $\mathcal{O}((1/\epsilon)^6\,\text{poly}(\log n))$ time. The number of rows of Φ equals $m = \mathcal{O}(\epsilon^{-2}\log n\log^*(\epsilon^{-1}))$.

The number of rows in Φ is more than that in [22] by merely a factor of $\mathcal{O}(\log^*(1/\epsilon))$, while the query time is sublinear for all small $\epsilon \geq n^{-1/7}$, a significant improvement upon the previous $\mathcal{O}(\epsilon^{-1}n\log n)$ runtime in [22].

Difference Between Deterministic and Explicit Schemes. To avoid confusion, we note the difference between 'deterministic' and 'explicit'. In the compressed sensing/sparse recovery literature 'deterministic' is also called 'for-all' or 'uniform', which means that a single matrix Φ suffices for the reconstruction of all vectors. 'Explicit' means that the matrix Φ can be constructed in time $\text{poly}(1/\epsilon, r, n)$. 'Strongly explicit' means that any entry of the matrix can be computed in time $\text{poly}(1/\epsilon, r, \log n)$. Hence, in this paper we show the existence of a

single matrix that allows reconstrction of all vectors, which we argue via the probabilistic method. The same holds for some of the schemes in [22].

In the strict turnstile model, we show that for the tail guarantee of $r = \lceil 1/\epsilon \rceil$, the matrix Φ can be made strongly explicit, with a mild increase in the number of rows, and the runtime can be improved polynomially. Note, however, the tail guarantee is with respect to $r = \lceil 1/\epsilon \rceil$ instead of $r = \lceil 1/\epsilon^2 \rceil$. We state our theorem below and omit the proof owing to space limitations.

▶ **Theorem 2** (ℓ_∞/ℓ_1, strict turnstile)**.** *There exists a strongly explicit matrix M of $O((1/\epsilon)^2 \log^3 n/\log^3(1/\epsilon))$ rows, which, given Mx in the strict turnstile model, allows us to find an $O(1/\epsilon)$-sparse vector $\hat{x}$ such that*

$$\|x - \hat{x}\|_\infty \leq \epsilon \|x_{-\lceil 1/\epsilon \rceil}\|_1,$$

in time $O((1/\epsilon)^3 \log^3 n/\log^3(1/\epsilon))$.

1.2 ℓ_1/ℓ_1 Sparse Recovery

In the ℓ_1/ℓ_1 sparse recovery problem, instead of the guarantee (1), the algorithm should output $\hat{x}$ such that

$$\|\hat{x} - x\|_1 \leq (1+\epsilon)\|x_{-k}\|_1. \tag{2}$$

It is known that any deterministic sketching algorithm requires $m = \Omega(\epsilon^{-2} + \epsilon^{-1} k \log(\epsilon n/k))$ rows of Φ [22], and the best known upper bound is $m = O(\epsilon^{-2} k \log(n/k))$ rows [12, 2, 10], suboptimal from the lower bound by only a logarithmic factor. However all these algorithms suffer from various defects: the algorithms in [12, 2] run in polynomial time in n, and that in [10] imposes a constraint on ϵ that precludes it from being a small constant when k is small. In this paper, we show that one can achieve sublinear runtime with the same number of rows for small k.

▶ **Theorem 3** (ℓ_1/ℓ_1)**.** *There exists a matrix $\Phi \in \mathbb{R}^{m \times n}$ such that, given Φx with $x \in \mathbb{R}^n$, we can find an $\mathcal{O}(k)$-sparse vector $\hat{x}$ satisfying (2) in $\mathcal{O}(k^3 \operatorname{poly}(1/\epsilon, \log n))$ time. The number of rows of Φ is $m = \mathcal{O}(\epsilon^{-2} k \log n)$.*

This result is the first sublinear time algorithm for all small $k \leq n^{0.3}$ with constant ϵ, while the algorithm in [10], using the same number of measurements, works for constant ϵ only when $n^{\delta'} \leq k \leq n^{1-\delta}$ (where $\delta, \delta' > 0$ are arbitrarily small constants), owing to its use of the list-decodable code. Combining the two results, we have solved the ℓ_1/ℓ_1 problem in sublinear time for all $k \leq n^{1-\delta}$ and constant ϵ.

▶ Remark. Our results heavily involve random hash functions, for which $\mathcal{O}(1/\epsilon)$- or $\mathcal{O}(k)$-wise independence would be sufficient. The space complexity of our algorithms is the same as the number of rows, unless stated otherwise.

2 Overview of Techniques

The main result on ℓ_∞/ℓ_1 combines different ideas from sparse recovery and heavy hitters literature. We first prove a result with the $\epsilon\|x_{-1/\epsilon}\|_1$ tail guarantee. We need a different, more careful construction of the weak system, akin to that in [10], which does not detect only a constant fraction of the heavy hitters, but a much larger fraction, as much as $(1 - \epsilon \log\log(1/\epsilon))$. One of our technical contributions and tools is the design of a more

general form of the weak system, which we then apply iteratively with carefully chosen parameters to recover all heavy hitters. We then iterate by subtracting the found heavy hitters, and try to find the remaining ones using a new matrix of the same number of resources (rows). Similar iteration techniques have been adopted in most combinatorial sparse recovery tasks, where the algorithms are allowed to miss even all heavy hitters if they are not large enough! Our ℓ_∞/ℓ_1 error guarantee, however, makes it prohibitive to miss small heavy hitters in later iterations. To that end, we heavily exploit the more abundant resource of $1/\epsilon^2$ rows in each iteration throughout, for which we pay a mild extra factor in the total number of rows. While in previous sparse recovery tasks the number of rows decreases across different iterations, this does not happen in our case. As aforementioned, we pay an additional $\log^*(1/\epsilon)$ factor as we shall have $\mathcal{O}(\log^*(1/\epsilon))$ iterations until all heavy hitters are recovered.

To obtain the stronger tail guarantee of $\epsilon\|x_{-1/\epsilon^2}\|_1$, we invoke additionally our ℓ_1/ℓ_1 algorithm and the point-query algorithm of [22]. We note that any sub-optimality in the number of rows of the ℓ_1/ℓ_1 linear sketch would yield a worse result for our main scheme, which forces us to obtain also an improved result for the ℓ_1/ℓ_1 problem. Our new weak system and the novel idea of using the iterative loop to satisfy the ℓ_∞/ℓ_1 guarantee may indicate new approaches to tackle heavy hitters tasks, and might be of interest beyond the scope of this paper. Our side-result on ℓ_1/ℓ_1 sparse recovery, is a combination of [10] and [16]. Specifically, one can avoid the Parvaresh-Vardy list-recoverable code that [10] employed, and use instead the clustering technique in [16], upon the two-layer hashing schemes and linking technique in [10]. This makes possible an improved result for ℓ_1/ℓ_1 that removes the restriction on ϵ the previous work of [10] was suffering from.

We remark that any improvement in the running time of the clustering algorithm of [16] immediately translates to improvement to ℓ_∞/ℓ_1 and ℓ_1/ℓ_1 schemes. More specifically, a near-quadratic or near-linear algorithm for that clustering would imply a near-quadratic or near-linear (in the number of rows of the sketching matrix) time algorithm for all three of our tasks. The current state of the art for that algorithm is $\tilde{\mathcal{O}}(N^3)$ runtime on a graph of N vertices, since the algorithm performs N calls to a routine that finds a Cheeger cut. We also remark that we could obtain our ℓ_∞/ℓ_1 result using explicit list-recoverable codes, such as Parvaresh-Vardy code, but this would lead to a slightly worse result than what we currently have.

In the strict turnstile model, we obtain a family of fully explicit matrices that solves the point query problem in sublinear time using the family of matrices from [22]. First, we show how to stengthen the guarantee obtained by the matrix in [22], achieving a stronger tail guarantee, and then we show how to recursively combine those explicit matrices to obtain a sketching algorithm that gives the ℓ_∞/ℓ_1 guarantee in sublinear time. Our result in the strict turnstile model, not only is fully explicit, but also has a better running time than the general turnstile model and gets also improved space for some regime of ϵ, namely $\epsilon \leq 2^{-\sqrt{\log n}}$. We note though that we obtain a weaker tail guarantee than in the general case, namely $r = 1/\epsilon$, instead of $1/\epsilon^2$. This weaker guarantee stems from the lack of fully explicit ℓ_1/ℓ_1 schemes with nearly optimal measurements, that can also answer queries in sublinear time.

3 Preliminaries

For a vector $x \in \mathbb{R}^n$, we define x_{-k} to be the vector obtained by zeroing out the largest k coordinates in magnitude, and $\text{supp}(x)$ to be the set of the non-zero coordinates of x. We also define $H(x, k)$ to be the index set of the largest k coordinates of x in magnitude. Thus,

$\mathrm{supp}(x_{-k}) \cap H(x,k) = \emptyset$. We also define $H(x,k,\epsilon) = \{i \in [n] : |x_i| \geq \epsilon/k \|x_{-k}\|_1\}$. We assume that the word size is $w = \Theta(\log n)$.

An error correcting code is a subset $\mathcal{C} \subseteq \Sigma^n$, where Σ is a finite set called alphabet and $|\mathcal{C}| = |\Sigma|^k$ for some $k \geq n$, together with an injective encoding map $\mathrm{enc} : \Sigma^k \to \mathcal{C}$ and a decoding map $\mathrm{dec} : \mathcal{C} \to \Sigma^k$. The parameter k is called the message length and n is called codeword length or block length. We say that an error correcting code can correct up to θ-fraction of errors if for any message $m \in \Sigma^k$ and any $x \in \Sigma^n$ such that the Hamming distance $d(\mathrm{enc}(m), x) \leq \theta n$, it holds that $\mathrm{dec}(x) = m$.

3.1 Two-layer Hashing Schemes

In this subsection we review the two-layer hashing scheme and the linking techniques used in [10], which will be the skeleton of construction for all our sparse recovery results.

First we recall some definitions, taken from [10], regarding bipartite expander, two-layer hashing and isolation of heavy hitters.

▶ **Definition 4** (bipartite expander)**.** An $(n, m, d, \ell, \epsilon)$-bipartite expander is a d-left-regular bipartite graph $G(L \cup R, E)$ where $|L| = n$ and $|R| = m$ such that for any $S \subseteq L$ with $|S| \leq \ell$ it holds that $|\Gamma(S)| \geq (1-\epsilon)d|S|$, where $\Gamma(S)$ is the neighbour of S (in R). When n and m are clear from the context, we abbreviate the expander as (ℓ, d, ϵ)-bipartite expander.

▶ **Definition 5** (one-layer hashing scheme)**.** The (N, B, d) (one layer) hashing scheme is the uniform distribution on the set of all functions $f : [N] \to [B]^d$. We write $f(x) = (f_1(x), \ldots, f_d(x))$, where f_i's are independent (N, B) hashing schemes.

Each instance of such a hashing scheme induces a d-left-regular bipartite graph with Bd right nodes. When N is clear from the context, we simply write (B, d) hashing scheme.

▶ **Definition 6** (two-layer hashing scheme)**.** An (N, B_1, d_1, B_2, d_2) (two-layer) hashing scheme is a distribution μ on the set of all functions $f : [N] \to [B_2]^{d_1 d_2}$ defined as follows. Let g be a random function subject to the (N, B_1, d_1) hashing scheme and $\{h_{i,j}\}_{i \in [d_1], j \in [d_2]}$ be a family of independent functions subject to the (B_1, B_2, d_2) hashing scheme which are also independent of g. Then μ is defined to be the distribution induced by the mapping

$$x \mapsto (h_{1,1}(g_1(x)), \ldots, h_{1,d_2}(g_1(x)), h_{2,1}(g_2(x)), \ldots, h_{2,d_2}(g_2(x)), \ldots, h_{d_1,1}(g_{d_1}(x)), \ldots, h_{d_1,d_2}(g_{d_1}(x))) .$$

Each instance of such a hashing scheme gives a $d_1 d_2$-left-regular bipartite graph of $B_2 d_1 d_2$ right nodes. When N is clear from the context, we simply write (B_1, d_1, B_2, d_2) hashing scheme. Conceptually we hash N elements into B_1 buckets and repeat d_1 times; these buckets will be referred to as first-layer buckets. In each of the d_1 repetitions, we hash B_1 elements into B_2 buckets and repeat d_2 times, those buckets will be referred to as second-layer buckets.

Bipartite expander graphs can be used as hashing schemes because of their isolation property.

▶ **Definition 7** (isolation property)**.** An $(n, m, d, \ell, \epsilon)$-bipartite expander G is said to satisfy the (ℓ, η, ζ)-isolation property if for any set $S \subset L(G)$ with $|S| \leq \ell$, there exists $S' \subset S$ with $|S'| \geq (1-\eta)|S|$ such that for all $x \in S'$ it holds that $|\Gamma(\{x\}) \setminus \Gamma(S \setminus \{x\})| \geq (1-\zeta)d$.

The following lemma shows that a random two-layer hashing satisfies a good isolation property with high probability. Previous works [25, 10] build sparse recovery systems upon this lemma.

▶ **Lemma 8** ([10]). *Let $\epsilon > 0$, $\alpha > 1$ and (N, B_1, d_1, B_2, d_2) be a two-layer hashing scheme with $B_1 = \Omega(\frac{k}{\zeta^\alpha \epsilon^{2\alpha}})$, $d_1 = \Omega(\frac{\alpha}{\alpha-1} \cdot \frac{1}{\zeta\epsilon} \frac{\log N}{\log(B_1/k)})$, $B_2 = \Omega(\frac{k}{\zeta\epsilon})$ and $d_2 = \Omega(\frac{1}{\zeta} \log \frac{B_1}{k})$. Then with probability $\geq 1 - 1/N^c$, the two-layer hashing scheme with parameters prescribed above gives an $(N, B_2 d_1 d_2, d_1 d_2, 4k, \epsilon)$ bipartite graph with the (L, ϵ, ζ)-isolation property, where $L = O(k/\epsilon)$.*

4 A Sublinear Time ℓ_1/ℓ_1 Algorithm

The result is almost immediate by replacing the list-recoverable code in [10] with the clustering algorithm in [16], which we now give a brief review. The overall algorithm is an iterative algorithm of $\Theta(\log k)$ iterations. Each iteration is called a *weak system*, which (i) recovers at least a constant fraction of the remaining heavy hitters (and hence all heavy hitters can be recovered in $\Theta(\log k)$ iterations) and (ii) introduces only a small amount of error (see, e.g. [25, 10]). The introduced error comes from two sources: the estimation error of those recovered heavy hitters and some small coordinates that can be safely ignored.

The following lemma is crucial in bounding the number of missed heavy hitters in iteration, modified from [10]. For completeness we include the proof in Appendix A.

▶ **Lemma 9.** *Let $\theta, \epsilon \in (0,1)$, $\delta \in (0, \frac{1}{2}]$ and $\beta, \zeta > 0$ such that $0 < \zeta < \delta - \frac{64\beta}{\theta}$. Suppose that G is a $(4s, d, \beta\epsilon)$-bipartite expander which satisfies the $(\frac{6}{\gamma\epsilon}, \frac{\epsilon\theta}{12}, \zeta)$-isolation property, where $\gamma \in [\frac{\theta}{s}, 1]$. Let $x \in \mathbb{R}^n$ be a vector which can be written as $x = y + z$, where y and z have disjoint supports, $|\operatorname{supp}(y)| \leq s$ and $\|z\|_1 \leq 3/2$. For each $i \in [n]$ define the multiset E_i as*

$$E_i = \left\{ \sum_{(u,v)\in E} x_u \right\}_{v \in \Gamma(\{i\})}.$$

Note that $|E_i| = d$ since it is a multiset. Then, for every $D \subset [n], |D| \leq 2s$, we have that

$$\left| \left\{ i \in D : |x_i - w| \geq \frac{\epsilon\gamma}{4} \text{ for at least } (1-\delta)d \text{ values } w \text{ in } E_i \right\} \right| \leq \frac{\theta}{\gamma}.$$

We remark that the construction of the weak system is similar to the partition setup in [16], where the linking information and the message block are 'absorbed' into the fashion coordinates are split into buckets so that recovering a heavy hitter in a bucket will automatically recover that information correctly instead of recovering the information from second-layer buckets with an error correcting code. In this paper, however, we opt for the two-layer construction for the 'for-all' guarantee, for the presentation would be simpler for our sparse recovery results as some auxiliary lemmata are already proved in [10].

We now present the precise statement of our weak system , which is central to our ℓ_1/ℓ_1 result. The proof is almost identical to that in [10] and is postponed to Appendix B.

▶ **Lemma 10** (Weak system). *Suppose that $s \leq \sqrt{n}$ and $\epsilon \in (0,1)$. There exist a linear sketch $\Phi \in \mathbb{R}^{m \times n}$ and an algorithm $\textsc{WeakSystem}(x, s, \epsilon)$ satisfying the following:*

- *For any vector $x \in \mathbb{R}^n$ that can be written as $x = y + z$, where y and z have disjoint supports, $|\operatorname{supp}(y)| \leq s$, $\|y\|_\infty \geq \epsilon/(2s)$ and $\|z\|_1 \leq 3/2$, given the measurements Φx, the decoding algorithm $\mathcal{D}$ returns $\hat{x}$ such that x admits the decomposition of*

  $$x = \hat{x} + \hat{y} + \hat{z},$$

  where $|\operatorname{supp}(\hat{x})| = s$, $|\operatorname{supp}(\hat{y})| \leq s/8$ and $\|\hat{z}\|_1 \leq \|z\|_1 + \epsilon/4$. Intuitively, $\hat{y}$ and $\hat{z}$ will be the head and the tail of the residual $x - \hat{x}$, respectively;
- *$m = \mathcal{O}(\epsilon^{-2} s \log n)$;*
- *$\mathcal{D}$ runs in $\mathcal{O}(s^3 \operatorname{poly}(1/\epsilon, \log n))$ time.*

5 A Sublinear Time ℓ_∞/ℓ_1 Algorithm

For ease of exposition and connection with the previous algorithms, we set $k = \lceil 1/\epsilon \rceil$ in this section.

First, we prove the following weaker theorem, and shall show how to bootstrap this theorem in order to obtain Theorem 1 in Section 5.1.

▶ **Theorem 11.** *There exists a linear sketch $\Phi \in \mathbb{R}^{m\times n}$ such that for every $x \in \mathbb{R}^n$, we can, given Φx, find an $\mathcal{O}(k)$-sparse vector $\hat{x}$ such that $\|x - \hat{x}\|_\infty \le \frac{1}{k}\|x_{-k}\|_1$ in $\mathcal{O}(k^6 \operatorname{poly}(\log n))$ time. The number of rows of Φ equals $m = \mathcal{O}(k^2 \log n \log^* k)$. The space needed to store (y, Φ) is $\mathcal{O}(k^2 \log n \cdot \log^* k \cdot \log\log k)$ words.*

We remark that this theorem is slightly weaker than our main result, namely Theorem 1, because the error is measured with respect to $\|x_{-k}\|_1$, and not with respect to $\|x_{-k^2}\|_1$. We are going to bootstrap Theorem 11 later.

Weak-Level System. Lemma 9 is a central argument for deterministic sparse recovery tasks. Previous works [25, 10] used the lemma with $\gamma = 1/s$ and constant $\theta \in (0, 1)$ to show that if we estimate every coordinate x_i to be the median of E_i and take the biggest $\Theta(s)$ estimates in magnitude, we shall miss at most $2s$ heavy hitters, upon which weak systems that miss a θ-fraction of heavy hitters were constructed. The overall algorithm makes sequential calls to weak systems with geometrically decreasing number of remaining heavy hitters. In our case, since we want the stronger ℓ_∞/ℓ_1 guarantee, we are not allowed to decrease geometrically the number of rows for the weak systems. But, with more allotted number of rows, we can recover much more than a constant fraction of heavy hitters by exploiting the power of θ. The full proof is postponed to Appendix C.

▶ **Lemma 12** (Weak system)**.** *Suppose that $w \le s \le k \le \sqrt{n}$ and $\eta \in (0, 1)$ is an arbitrarily small constant. There exist a linear sketch $\Phi \in \mathbb{R}^{m\times n}$ and an algorithm* WeakSystem *(x, k, s, w) satisfying the following:*

- *For any vector $x \in \mathbb{R}^n$ that can be written as $x = y + z$, where y and z have disjoint supports, $|\operatorname{supp}(y)| \le s$, $\|y\|_\infty \ge 1/(2k)$ and $\|z\|_1 \le 2 - s/k$, given the sketch Φx, the decoding algorithm $\mathcal{D}$ returns $\hat{x}$ such that x admits the decomposition of*

  $$x = \hat{x} + \hat{y} + \hat{z},$$

  where $|\operatorname{supp}(\hat{x})| = s$, $\|(x - \hat{x})_{\operatorname{supp}(\hat{x})}\|_\infty \le \frac{1}{2k}$, $|\operatorname{supp}(\hat{y})| \le \sqrt{sw}$ and $\|\hat{z}\|_1 \le 2 - \frac{s}{2k}$. Intuitively, $\hat{y}$ and $\hat{z}$ will be the head and the tail of the residual $x - \hat{x}$, respectively.
- *$m = \mathcal{O}(\frac{k^2}{w} \log n)$,*
- *$\mathcal{D}$ runs in $\mathcal{O}(k^6 \operatorname{poly}(\log n))$ time.*

Proof Sketch. We only specify the parameters of the two-layer hashing and the application of Lemma 9 below. We instantiate the two-layer hashing and the encoding scheme as in Section 3.1, where $\alpha \in (1, 2)$, $B_1 = \Theta(k^{2\alpha})$, $d_1 = \Theta(\frac{k}{\sqrt{sw}} \frac{\log n}{\log(B_1/s)})$, $B_2 = \Theta(k\sqrt{s/w})$ and $d_2 = \Theta(\log(B_1/s))$. By Lemma 8 (to see the conditions hold, replace k with s and ϵ with $k/\sqrt{sw}$), we can find a two-layer hashing with these prescribed parameters which satisfies $(\Theta(k), d_1 d_2, \frac{\sqrt{sw}}{k})$-expansion property and $(\Theta(k), \frac{\sqrt{sw}}{k}, \Theta(1))$-isolation property. Invoking Lemma 9 with $\delta = \Theta(\sqrt{sw}/k)$, $\theta = \Theta(\sqrt{sw}/k)$, $\epsilon = 1$ and $\gamma = 1/k$, and following the argument in [10, Section 4.1], we have good estimates for all but at most $\Theta(\theta/\gamma) = \sqrt{sw}$ heavy hitters (elements in $\operatorname{supp}(y)$). We are able to recover those heavy hitters as argued in [10] with the clustering algorithm from [16]. ◀

Algorithm 1 Overall algorithm for ℓ_∞/ℓ_1 sparse recovery. In the pseudocode below, $v^{(i,r)}$ as an argument of WeakSystem is understood to be restricted to the corresponding coordinates.

Input: sketching matrix Φ, sketch $v = \Phi x$, sparsity parameter k
Output: $\hat{x}$ that approximates x with ℓ_∞/ℓ_1 guarantee

$v^{(0,1)} \leftarrow v$
$k_1 \leftarrow k$
while $k_r > 4$ **do** ▷ $\mathcal{O}(\log^* k)$ rounds
  $i \leftarrow 0$
  while $k_r^{2^{-i}} \geq \max\{(i+1)^2, 4(1+\frac{1}{i+1})^4\}$ **do** ▷ $\mathcal{O}(\log\log k_r)$ steps
    $s_{i,r} \leftarrow (i+1)^2 k_r^{2^{-i}}$
    $w_{i,r} \leftarrow (i+1)^2$
    $\hat{x}^{(i,r)} \leftarrow \text{WeakSystem}(\Phi^{(i,r)}, v^{(i,r)}, k, s_{i,r}, w_{i,r})$
    $v^{(i+1,r)} \leftarrow v^{(i,r)} - \Phi\hat{x}^{(i,r)}$
    $i \leftarrow i+1$
  end while
  $i_r^* \leftarrow i-1$
  $s_{i,r} \leftarrow (s_{i_r^*,r})^{2^{-(i-i_r^*-1)}}$
  while $s_{i,r} \geq \max\{4, \log\log k_r\}$ **do** ▷ $\mathcal{O}(1)$ steps
    $w_{i,r} \leftarrow 1$
    $\hat{x}^{(i,r)} \leftarrow \text{WeakSystem}(\Phi^{(i,r)}, v^{(i,r)}, k, s_{i,r}, w_{i,r})$
    $v^{(i+1,r)} \leftarrow v^{(i,r)} - \Phi\hat{x}^{(i,r)}$
    $i \leftarrow i+1$
    $s_{i,r} \leftarrow (s_{i_r^*,r})^{2^{-(i-i_r^*-1)}}$
  end while
  $i_r^+ \leftarrow i - i_r^* - 1$
  $v^{(0,r+1)} \leftarrow v^{(i,r)}$
  $k_{r+1} \leftarrow s_{i,r}$
  $r \leftarrow r+1$
end while
$\hat{x}_{\text{final}} \leftarrow \text{WeakSystem}(\Phi^{(0,r)}, v^{(0,r)}, k, 4, 1/5)$ ▷ last round
return $\hat{x} \leftarrow \hat{x}_{\text{final}} + \sum_{r,i} \hat{x}^{(i,r)}$

Construction of Measurement Matrix. Now we construct the sketch for Theorem 11. The main idea is to apply the weak system (Lemma 12) repeatedly. We form our linear sketch Φ as illustrated below and present our recovery algorithm in Algorithm 1.

$$\Phi = \begin{bmatrix} \Phi_1 \\ \Phi_2 \\ \vdots \\ \Phi_R \\ \Phi_{\text{final}} \end{bmatrix}, \qquad \text{where } \Phi_r = \begin{bmatrix} \Phi^{(1,r)} \\ \vdots \\ \Phi^{(i_r^*,r)} \\ \Phi^{(i_r^*+1,r)} \\ \vdots \\ \Phi^{(i_r^*+i_r^+,r)} \end{bmatrix}, r = 1, \ldots, R.$$

Here

- the overall Φ is the vertical concatenation of $R+1$ matrices and every layer, except the last one, is further a concatenation of $i_r^* + i_r^+$ matrices, where $R = \Theta(\log^* k)$ and i_r^*, i_r^+

are computed as in Algorithm 1;
- the i-th layer in Φ_r, namely $\Phi_{i,r}$, is the sketching matrix for the weak system (Lemma 12) with parameters $s = s_{i,r}$ and $w = w_{i,r}$, the values of which are as assigned in Algorithm 1;
- the last layer of Φ, namely Φ_{final}, is the sketching matrix for the weak system with parameters $s = 4$ and $w = 1/5$.

Overall there are $i_r^* + i_r^+ + 1$ iterations in the algorithm and each iteration corresponds to one block of Φ. There are k heavy hitters at the beginning. Each iteration reduces the number of remaining heavy hitters to almost its square root and hence in $O(\log\log k)$ iterations the number of remaining heavy hitters will be reduced to a constant, and those heavy hitters will be all recovered in the last iteration. In order to minimize the number of measurements, in each of the first i_r^* iterations, the number of remaining heavy hitters is reduced to slightly bigger than its square root, and in each of the next i_r^+ iterations, to exactly the square root.

The parameters $s_{i,r}, w_{i,r}, i_r^*, i_r^+$ may seem adaptive at the first glance, but they in fact do not depend on the input x and depend only on the sparsity parameter k and can thus be pre-computed. The whole algorithm is non-adaptive.

Proof of Theorem 11. We only provide a sketch of the proof below and leave the full proof to Appendix D.

Proof Sketch. Without loss of generality, assume that $\|x_{-k}\|_1 = 1$. We shall apply Lemma 12 repeatedly to obtain a sequence of vectors $\hat{x}^{(i,r)}$, which admit decompositions $x^{(i,r)} = x - \hat{y}^{(i,r)} - \hat{z}^{(i,r)}$. We can show inductively that the loops invariants (I) below, parametrized by (i, r), are satisfied at the beginning on each while loop from Line 5 to 11 in Algorithm 1 and the loop invariants (II) below are satisfied at the beginning on each while loop from Line 14 to 20.

$$\text{(I)} \begin{cases} |\operatorname{supp}(\hat{y}^{(i,r)})| \le s_{i,r} := (i+1)^2 k_r^{2^{-i}}, \\ \|\hat{z}^{(i,r)}\|_1 \le 2 - s_{i,r}/k; \end{cases} \qquad \text{(II)} \begin{cases} |\operatorname{supp}(\hat{y}^{(i,r)})| \le s_{i,r} := (s_{i_r^*,r})^{2^{-(i-i_r^*-1)}}, \\ \|\hat{z}^{(i,r)}\|_1 \le 2 - s_{i,r}/k. \end{cases}$$

When the algorithm runs into Line 25, that is, when there are at most 4 heavy hitter left, we shall recover all of them in one call to the weak system.

The total number of rows and runtime, etc., follow from direct calculations. ◀

5.1 Getting the Final Result

We now show how to combine the ℓ_1/ℓ_1 scheme with the ℓ_∞/ℓ_1 scheme to obtain the main result of the paper. For completeness, we restate the main theorem with the substitution of $k = \lceil 1/\epsilon \rceil$.

▶ **Theorem 1** (rephrased). *There exists a linear sketch $\Phi \in \mathbb{R}^{m\times n}$ such that for every $x \in \mathbb{R}^n$, we can, given Φx, find an $\mathcal{O}(k)$-sparse vector $\hat{x}$ such that $\|x - \hat{x}\|_\infty \le (1/k)\|x_{-k^2}\|_1$ in $\mathcal{O}(k^6 \operatorname{poly}(\log n))$ time. The sketch length is $m = \mathcal{O}(k^2 \log n \log^* k)$ and the space needed to store (y, Φ) is $\mathcal{O}(k^2 \log n \cdot \log^* k \cdot \log\log k)$ words.*

We shall need the following lemma from [22].

▶ **Lemma 13** (Point Query [22]). *There exists a matrix $C \in \mathbb{R}^{m\times n}$ with $m = \mathcal{O}(k^2 \log n)$ rows, such that given $y = Cx$ and $i \in [n]$, it is possible to find in $\mathcal{O}(k \log n)$ time a value $\hat{x}_i$ such that $|x_i - \hat{x}_i| \le (1/k)\|x_{[n]\setminus\{i\}}\|_1$.*

The construction of C given in [22] includes taking C to be a Johnson-Lindenstrauss Transform matrix for the set of points $\{0, e_1, \ldots, e_n\}$, where $e_1, \ldots, e_n$ is the canonical basis of $\mathbb{R}^n$.

We now give a sketch of the proof of Theorem 1 and leave the full proof to Appendix E.

Proof of Theorem 1 (Sketch). We first pick a matrix A using Lemma 3, setting the sparsity parameter to k^2 and $\epsilon = 1$. We also pick a matrix B satisfying the guarantees of Theorem 11, with sparsity $6k$, and a matrix C using Lemma 13 with sparsity parameter $6k$. Our sketching matrix Φ is the vertical concatenation of A, B and C.

We first run the algorithm on Ax to obtain an $\mathcal{O}(k^2)$-sparse vector z such that $\|x - z\|_1 \leq 2\|x_{-k^2}\|_1$. Then we form $B(x - z)$ and using the query algorithm for B, we find an $\mathcal{O}(k)$-sparse vector w such that $\|(x - z) - w\|_\infty \leq \frac{1}{2k}\|x - z\|_1 \leq \frac{1}{k}\|x_{-k^2}\|_1$. It then follows that $\left\{i \in [n] : |x_i| > \frac{1}{k}\|x_{-k^2}\|_1\right\} \subseteq \operatorname{supp}(z) \cup \operatorname{supp}(w)$ and so it suffices to estimate x_i, for every $i \in \operatorname{supp}(z) \cup \operatorname{supp}(w)$, up to $(1/k)\|x_{-k^2}\|_1$ error, which we can do exactly as in [22].

The total number of rows and runtime, etc., follow from direct calculations. ◀

6 Conclusion and Open Problems

In this work, we present the first algorithm for finding ℓ_1 heavy hitters (ℓ_∞/ℓ_1 guarantee) deterministically in sublinear time, up to an $\mathcal{O}(\log^*(1/\epsilon))$ factor in the number of measurement from the best superlinear-time algorithm. It still remains to improve the dependence on ϵ in the running time, ideally to $\mathcal{O}(\epsilon^{-2}\operatorname{poly}(\log n))$. The problem could first be approached in the strict turnstile model, where it is possible to avoid the heavy machinery of list-recoverable codes or the clustering algorithm of [16]. Another open problem is to find (fully) explicit constructions that allows a sublinear-time decoding with the number of rows near $\mathcal{O}(\epsilon^{-2}\log n)$ in the strict turnstile model. In the general turnstile model, our current understanding and techniques suggest that an explicit scheme would require an explicit construction of expanders or lossless condensers, together with list-recoverable codes with nearly optimal encoding and decoding time, constructions that are currently out of reach. In conclusion, we hope that our work will ignite further work in the field, and towards the resolution of some of these questions.

References

1. Alexandr Andoni, Robert Krauthgamer, and Krzysztof Onak. Streaming algorithms via precision sampling. In *Proceedings of the 2011 IEEE 52Nd Annual Symposium on Foundations of Computer Science*, FOCS '11, pages 363–372, Washington, DC, USA, 2011. IEEE Computer Society.
2. R. Berinde, A. C. Gilbert, P. Indyk, H. Karloff, and M. J. Strauss. Combining geometry and combinatorics: A unified approach to sparse signal recovery. In *2008 46th Annual Allerton Conference on Communication, Control, and Computing*, pages 798–805, Sept 2008. doi:10.1109/ALLERTON.2008.4797639.
3. Radu Berinde, Piotr Indyk, Graham Cormode, and Martin J. Strauss. Space-optimal heavy hitters with strong error bounds. *ACM Trans. Database Syst.*, 35(4):26:1–26:28, 2010. doi:10.1145/1862919.1862923.
4. Albert Cohen, Wolfgang Dahmen, and Ronald DeVore. Compressed sensing and best k-term approximation. *Journal of the American mathematical society*, 22(1):211–231, 2009.
5. 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.

6 S Ben Fred, Thomas Bonald, Alexandre Proutiere, Gwénaël Régnié, and James W Roberts. Statistical bandwidth sharing: a study of congestion at flow level. In *ACM SIGCOMM Computer Communication Review*, volume 31, pages 111–122. ACM, 2001.

7 Sumit Ganguly. Lower bounds on frequency estimation of data streams. *Lecture Notes in Computer Science*, 5010:204–215, 2008.

8 A. C. Gilbert, M. J. Strauss, J. A. Tropp, and R. Vershynin. One sketch for all: Fast algorithms for compressed sensing. In *Proceedings of the Thirty-ninth Annual ACM Symposium on Theory of Computing*, STOC '07, pages 237–246, New York, NY, USA, 2007. ACM.

9 Anna C Gilbert, Yi Li, Ely Porat, and Martin J Strauss. Approximate sparse recovery: optimizing time and measurements. *SIAM Journal on Computing*, 41(2):436–453, 2012.

10 Anna C. Gilbert, Yi Li, Ely Porat, and Martin J. Strauss. For-all sparse recovery in near-optimal time. *ACM Trans. Algorithms*, 13(3):32:1–32:26, 2017.

11 Nicholas J. A. Harvey, Jelani Nelson, and Krzysztof Onak. Sketching and streaming entropy via approximation theory. In *Proceedings of the 2008 49th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '08, pages 489–498, Washington, DC, USA, 2008. IEEE Computer Society.

12 P. Indyk and M. Ruzic. Near-optimal sparse recovery in the l1 norm. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 199–207, Oct 2008. doi:10.1109/FOCS.2008.82.

13 T. S. Jayram and D. P. Woodruff. The data stream space complexity of cascaded norms. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 765–774, Oct 2009.

14 Hossein Jowhari, Mert Sağlam, and Gábor Tardos. Tight bounds for lp samplers, finding duplicates in streams, and related problems. In *Proceedings of the Thirtieth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '11, pages 49–58, New York, NY, USA, 2011. ACM.

15 Daniel M. Kane and Jelani Nelson. Sparser johnson-lindenstrauss transforms. *J. ACM*, 61(1):4:1–4:23, 2014. doi:10.1145/2559902.

16 Kasper Green Larsen, Jelani Nelson, Huy L Nguyen, and Mikkel Thorup. Heavy hitters via cluster-preserving clustering. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 61–70. IEEE, 2016.

17 Yi Li, Huy L. Nguyen, and David P. Woodruff. Turnstile streaming algorithms might as well be linear sketches. In *Proceedings of the Forty-sixth Annual ACM Symposium on Theory of Computing*, STOC '14, pages 174–183, New York, NY, USA, 2014. ACM. doi:10.1145/2591796.2591812.

18 Gurmeet Singh Manku and Rajeev Motwani. Approximate frequency counts over data streams. In *Proceedings of the 28th International Conference on Very Large Data Bases*, VLDB '02, pages 346–357. VLDB Endowment, 2002. URL: http://dl.acm.org/citation.cfm?id=1287369.1287400.

19 Ahmed Metwally, Divyakant Agrawal, and Amr El Abbadi. *Efficient Computation of Frequent and Top-k Elements in Data Streams*, pages 398–412. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

20 Morteza Monemizadeh and David P. Woodruff. 1passs relative-error lp-sampling with applications. In *Proceedings of the Twenty-first Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '10, pages 1143–1160, Philadelphia, PA, USA, 2010. Society for Industrial and Applied Mathematics.

21 Tatsuya Mori, Ryoichi Kawahara, Shozo Naito, and Shigeki Goto. On the characteristics of internet traffic variability: Spikes and elephants. *IEICE TRANSACTIONS on Information and Systems*, 87:2644–2653, 2004.

22 Jelani Nelson, Huy L. Nguyên, and David P. Woodruff. On deterministic sketching and streaming for sparse recovery and norm estimation. 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 627–638, 2012.

23 Konstantina Papagiannaki, Nina Taft, S Bhattacharya, Patrick Thiran, Kave Salamatian, and Christophe Diot. On the feasibility of identifying elephants in internet backbone traffic. *Sprint Labs, Sprint ATL, Tech. Rep. TR01-ATL-110918*, 2001.

24 Mark S. Pinsker. On the complexity of a concentrator. In *Proceedings of 7th International Teletraffic Conference*, 1973.

25 Ely Porat and Martin J. Strauss. Sublinear time, measurement-optimal, sparse recovery for all. In *Proceedings of the Twenty-third Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '12, pages 1215–1227, 2012.

26 Liang Yang, Bryan Ng, and Winston KG Seah. Heavy hitter detection and identification in software defined networking. In *Computer Communication and Networks (ICCCN), 2016 25th International Conference on*, pages 1–10. IEEE, 2016.

27 Yin Zhang, Lee Breslau, Vern Paxson, and Scott Shenker. On the characteristics and origins of internet flow rates. In *ACM SIGCOMM Computer Communication Review*, volume 32, pages 309–322. ACM, 2002.

28 Yu Zhang, BinXing Fang, and YongZheng Zhang. Identifying heavy hitters in high-speed network monitoring. *Science China Information Sciences*, 53(3):659–676, 2010.

A Proof of Lemma 9

Proof. Suppose that $|D| > \theta s$, otherwise the result holds automatically. Assume that $|x_1| \geq |x_2| \geq \cdots \geq |x_n|$. Let $T = D \cup \{i : |x_i| \geq \epsilon\gamma/4\}$, then $t := |T| \leq \|z\|_1/(\epsilon\gamma/4) + |D| \leq 6/(\epsilon\gamma)$.

Note that $|x_{t+1}| \leq \epsilon\gamma/4$. Taking $\alpha = 2$ in [10, Lemma 3.3], we know that

$$\|(\Phi(x - x_{[t]}))_{\Gamma(D)}\|_1 \leq 4 \cdot \beta\epsilon d \left(\frac{3}{2} + 2s \cdot \frac{\epsilon\gamma}{4}\right) \leq 8\beta\epsilon d.$$

By the isolation property, there are at most $\frac{6}{\epsilon\gamma} \cdot \frac{\epsilon\theta}{12} = \frac{\theta}{2\gamma}$ elements in T which are not isolated in at least $(1-\zeta)d$ nodes from other elements in T. This implies that at least $\theta/(2\gamma)$ elements in D are isolated in at least $(1-\zeta)d$ nodes from other elements in T.

A decoy at position i receives at least $\epsilon\gamma/4$ noise in at least $(\beta-\zeta)d$ isolated nodes of $\Gamma(\{i\})$, hence in total, a decoy element receives at least $\epsilon\gamma(\beta-\zeta)d/4$ noise. Therefore at least $\theta/(2\gamma)$ decoys overall should receive noise at least

$$\frac{\epsilon\gamma(\beta-\zeta)d}{4} \cdot \frac{\theta}{2\gamma} > 8\beta\epsilon d \geq \|(\Phi(x - x_{[t]}))_{\Gamma(D)}\|_1,$$

which is a contradiction. Therefore there are at most θs decoys. ◀

B Proof of Lemma 10

Before presenting the proof, we sketch our setup of message encoding. Let $\mathrm{enc} : \{0,1\}^{\log n} \to \{0,1\}^{\mathcal{O}(\log n)}$ be an error-correcting code that corrects a constant fraction of errors in linear time. For notational convenience, let $m_i = \mathrm{enc}(i)$, the codeword for the binary representation of i. Furthermore, we break m_i into d_1 blocks of length $\Theta((\log n)/d_1)$ each, say, $m_i = (m_{i,1}, \ldots, m_{i,d_1})$.

Let G be a Δ-regular edge-expander graph on d_1 vertices, where Δ is an absolute constant (we may assume that d_1 is even and such edge expander exists by [24]). Let j be a node in G and denote its neighbours by $\Gamma_1(j), \ldots, \Gamma_\Delta(j)$. Let $idx(r, i)$ ($r \in [d_1]$ and $i \in [n]$) denote the index of the bucket where i is hashed in the r-th first-layer repetition. Construct the message

$$\bar{m}_{i,r} = m_{i,r} \circ idx(\Gamma_1(r), i) \circ \cdots \circ idx(\Gamma_\Delta(r), i), \qquad i \in [n], r \in [d_1],$$

where $\circ$ denotes concatenation of strings and $idx(\cdot, \cdot)$ is understood as its binary representation of $\log(B_1)$ bits.

Now for each index i we have d_1 blocks of message $\bar{m}_{i,1}, \ldots, \bar{m}_{i,d_1}$. We can protect each block using a constant-rate error correcting code which tolerates a constant fraction of error and decodes in polynomial time, so that if we can recover a fraction of $\bar{m}_{i,r}$ we can recover the entire message $\bar{m}_{i,r}$ efficiently. The high-level idea is then to recover a good fraction of $\{\bar{m}_{i,r}\}_{r\in[d]}$ for a good fraction of heavy hitters i, so that we can recover m_i using the linking information embedded in $\bar{m}_{i,r}$ and the clustering algorithm in [16]. Finally we decode m_i to obtain the corresponding index i.

Proof of Lemma 10. We follow the construction and the argument as in [10]. We instantiate the two-layer hashing and the encoding scheme as in Section 3.1, where $\alpha \in (1, 2)$, $B_1 = \Theta(s^\alpha/\epsilon^{2\alpha})$, $d_1 = \Theta(\epsilon^{-1}\frac{\log n}{\log(B_1/s)})$, $B_2 = \Theta(s/\epsilon)$ and $d_2 = \Theta(\log(B_1/s))$. By Lemma 8 we can find a two-layer hashing with these prescribed parameters which satisfies $(4s, d_1d_2, O(\epsilon))$-expansion property and $(O(s/\epsilon), O(\epsilon), \Theta(1))$-isolation property. It is also easy to verify that the length of each message block $\bar{m}_{i,r}$ is $L = \Theta(\log(B_1/s)) + \Delta\log(B_1) \le d_2/2$ if we choose d_2 large enough. We can use two second-layer measurements to encode 1 bit of message by replacing a single entry a in the measurement matrix with a 2×1 block of $\binom{a}{0}$ or $\binom{0}{a}$, depending on the bit to encode. To decode the bit, suppose that the two corresponding measurements are $\binom{a}{b}$ and we convert them back to 0 if $|a| < |b|$ and 1 if $|a| \ge |b|$. When the heavy hitter is isolated and the noise is small in a bucket, the bit is expected to be the corresponding bit in the message for that heavy hitter.

Next we shall show that we can recover most bits of the message for at least a large constant fraction of heavy hitters. Invoking Lemma 9 with $\delta = O(1)$, $\theta = O(1)$, $\beta = O(1)$ and $\gamma = 1/s$, and following the argument in [10, Section 4.1], we have good estimates for all but at most $s/8$ heavy hitters (elements in $\mathrm{supp}(y)$). Call those heavy hitters *well-estimated*. The two-layer hashing eventually hashes n coordinates into B_2 buckets and repeat d_1d_2 times, and we know that each well-estimated heavy hitter i receives small noise in at least $(1-\delta)d_1d_2$ repetitions. This implies that there exist δ_1 and δ_2 such that for each well-estimated heavy hitter i, there exist $(1-\delta_1)d_1$ first-layer repetitions such that in each such first-layer repetition r the heavy hitter i receives small noise in at least $(1-\delta_2)d_2$ second-layer repetitions. For each such pair (i, r), we can recover at least $(1-\delta_2)$ fraction of the message $\bar{m}_{i,r}$, and if we protect $\bar{m}_{i,r}$ using a constant-rate error-correcting code (e.g. Reed-Solomon code) that can tolerate up to δ_2 fraction of error, we shall recover $\bar{m}_{i,r}$ in entirety. To summarize, for each well-estimated heavy hitter i, we can recover $\bar{m}_{i,r}$ for at least $(1-\delta_1)d_1$ values of $r \in [d_1]$. We note that δ_1 can made arbitrarily small by adjusting the constants in the two-layer construction and making δ arbitrarily small.

Now we construct the chunk graph as in [16]. The chunk graph has B_1d_1 nodes, indexed by pairs (b, r) for $b \in [B_1]$ and $r \in [d_1]$. For each bucket b in the first-repetition r, we recover a message of length L, break it up into blocks of the same structure as in $\hat{m}$ and extract the linking information $q_1(b, r), \ldots, q_\Delta(b, r)$. We say in $\tilde{G}$ the node (b, r) makes suggestion to connect to $(q_\ell(b, r), \Gamma_\ell(r))$, and we add an edge if both endpoints suggest each other. By the argument in [16, Lemma 2], a well-estimated heavy hitter i corresponds to an ϵ_0-spectral

cluster of $\tilde{G}$ for some small $\epsilon_0 > 0$. The spectral clustering algorithm ([16, Theorem 1]) will find all those spectral clusters, recovering a constant fraction of message m_i and enabling us to identify the index i of the heavy hitter. In each first-layer we retain the bucket of magnitude at least $\epsilon/(4s)$ so there are $O(s/\epsilon)$ buckets, and we therefore have a candidate list of size $\mathcal{O}(s/\epsilon)$ which misses at most $s/8$ heavy hitters. Finally we evaluate every candidate and retain the biggest s ones (in magnitude).

Each recovered coordinate is estimated to within $\epsilon/(4s)$, thus

$$\|\hat{z}\|_1 \leq \|z\|_1 + |\operatorname{supp}(\hat{x})| \cdot \frac{\epsilon}{4s} \leq \|z\|_1 + \frac{\epsilon}{4}.$$

The total number of measurements is $B_2 d_1 d_2 = \mathcal{O}(\epsilon^{-2} s \log n)$. For each first-layer repetition, we enumerate all coordinates in the bucket of size B_1, and decode the associated message (which is of length d_2), which takes time $\mathcal{O}(B_1 \operatorname{poly}(d_2)) = \mathcal{O}(s^\alpha \operatorname{poly}(1/\epsilon, \log n))$. We then run the spectral clustering algorithm on a graph of size $O(s/\epsilon \cdot d_1)$ in time $\tilde{\mathcal{O}}((s/\epsilon \cdot d_1)^3) = \mathcal{O}(s^3 \operatorname{poly}(1/\epsilon, \log n))$. To obtain the indices of the candidates, We decode $\mathcal{O}(s/\epsilon \cdot d_1) = \mathcal{O}(\epsilon^{-2} s \log n)$ messages, and the decoding algorithm on each $\Theta(\log n)$-bit-long m_i with a constant fraction corruption runs in time $O(\operatorname{poly}(\log n))$. Lastly we estimate each of the candidates and retain the biggest s ones, which takes time $\mathcal{O}((s/\epsilon) d_1 \cdot B_2 d_1 d_2) = \mathcal{O}(s^2 \operatorname{poly}(1/\epsilon, \log n))$. The overall runtime is dominated by the clustering algorithm and is therefore $\mathcal{O}(s^3 \operatorname{poly}(1/\epsilon, \log n))$. ◀

C Proof of Lemma 12

We follow the argument in [10]. We instantiate the two-layer hashing and the encoding scheme as in Section 3.1, where $\alpha \in (1, 2)$, $B_1 = \Theta(k^{2\alpha})$, $d_1 = \Theta(\frac{k}{\sqrt{sw}} \frac{\log n}{\log(B_1/s)})$, $B_2 = \Theta(k\sqrt{s/w})$ and $d_2 = \Theta(\log(B_1/s))$. By Lemma 8 (to see the conditions hold, replace k with s and ϵ with $k/\sqrt{sw}$), we can find a two-layer hashing with these prescribed parameters which satisfies $(\Theta(k), d_1 d_2, \frac{\sqrt{sw}}{k})$-expansion property and $(\Theta(k), \frac{\sqrt{sw}}{k}, \Theta(1))$-isolation property. The constants in the Θ-notations above all depend on η. It is also easy to verify that the length of each message block $\bar{m}_{i,r}$ is $L = \Theta(\log(B_1/s)) + \Delta \log(B_1) \leq d_2$ if we choose d_2 large enough.

Invoking Lemma 9 with $\delta = \Theta(\sqrt{sw}/k)$, $\theta = \Theta(\sqrt{sw}/k)$, $\epsilon = 1$ and $\gamma = 1/k$, and following the argument in [10, Section 4.1], we have good estimates for all but at most $\Theta(\theta/\gamma) = \sqrt{sw}$ heavy hitters (elements in $\operatorname{supp}(y)$). Call those heavy hitters *well-estimated*. Following the same argument as in the proof of Lemma 12, we can, for each well-estimated heavy hitter i, recover $\bar{m}_{i,r}$ for at least $(1 - \delta_1)d_1$ values of $r \in [d_1]$. We note that δ_1 can made arbitrarily small by adjusting the constants in the two-layer construction and making δ arbitrarily small.

We construct the chunk graph $\tilde{G}$ as in [16]. By the argument in [16, Lemma 2], a well-estimated heavy hitter i corresponds to an ϵ_0-spectral cluster of $\tilde{G}$ for some small $\epsilon_0 > 0$. The spectral clustering algorithm ([16, Theorem 1]) will find all those spectral clusters, recovering a constant fraction of message m_i and enabling us to identify the index i of the heavy hitter. In each first-layer we retain the bucket of magnitude at least $1/(4k)$ so there are $O(k)$ buckets, and we therefore have a candidate list of size $\mathcal{O}(k)$ which misses at most $\sqrt{sw}$ heavy hitters. Finally we evaluate every candidate and retain the biggest s ones (in magnitude).

Each recovered coordinate is estimated to within $\epsilon\gamma/4 \leq 1/(4k)$, thus

$$\|\hat{z}\|_1 \leq \|z\|_1 + \frac{|\operatorname{supp}(\hat{x})|}{4k} \leq 2 - \frac{s}{k} + \frac{s}{2k} \leq 2 - \frac{s}{2k}.$$

The total number of rows is $B_2 d_1 d_2 = \mathcal{O}(\frac{k^2}{w} \log n)$. For each first-layer repetition, we enumerate all coordinates in the bucket of size B_1, and decode the associated message (which

is of length d_2), which takes time $\mathcal{O}(B_1 \operatorname{poly}(d_2)) = \mathcal{O}(k^{2\alpha} \operatorname{poly}(\log k))$. We then run the spectral clustering algorithm on a graph of size $\mathcal{O}(kd_1)$ in time $\mathcal{O}((kd_1)^3) = \mathcal{O}(k^6 \operatorname{poly}(\log n))$. To obtain the indices of the candidates, We decode $\mathcal{O}(kd_1) = \mathcal{O}(k^2 \log n)$ messages, and the decoding algorithm on each $\Theta(\log n)$-bit-long m_i with a constant fraction corruption runs in time $O(\operatorname{poly}(\log n))$. Lastly we estimate each of the candidates and retain the biggest s ones, which takes time $\mathcal{O}(kd_1 \cdot B_2 d_1 d_2) = \mathcal{O}(k^4 \log^2 n)$. The overall runtime is dominated by the clustering algorithm and is therefore $\mathcal{O}(s^6 \operatorname{poly}(\log n)) = \mathcal{O}(k^6 \operatorname{poly}(\log n))$.

D Proof of Theorem 11

Without loss of generality, assume that $\|x_{-k}\|_1 = 1$. We shall apply Lemma 12 repeatedly to obtain a sequence of vectors $\hat{x}^{(i,r)}$, which admit decompositions $x^{(i,r)} = x - \hat{y}^{(i,r)} - \hat{z}^{(i,r)}$. Consider the following loop invariants, parametrized by (i, r), at the beginning of the i-th step in the r-th round:

$$\begin{aligned} |\operatorname{supp}(\hat{y}^{(i,r)})| &\le s_{i,r} := (i+1)^2 k_r^{2^{-i}} \\ \|\hat{z}^{(i,r)}\|_1 &\le 2 - s_{i,r}/k \end{aligned} \tag{3}$$

We claim that the loop invariants above hold for $(0, r)$ for $r = O(\log^* k)$. The base case is $(0, 0)$ and the loop invariants holds trivially. Suppose that the loop invariants hold for (i, r), we shall show that it holds for $(i, r+1)$ whenever $r \le r_0$ for some $r_0 = O(\log^* k)$.

To prove the inductive step w.r.t. r, we consider an inductive proof w.r.t. i for a fixed r. For the inductive step, if $i < i_r^*$ we apply Lemma 12 with $w_{i,r} = (i+1)^2$ and $s_{i,r} = (i+1)^2 k_r^{2^{-i}}$. We then get that

$$|\operatorname{supp}(\hat{y}^{(i,r)})| \le \sqrt{s_{i,r} w_{i,r}} = (i+1)^2 k_r^{2^{-(i+1)}} \le s_{i+1,r},$$

and

$$\|\hat{z}^{(i,r)}\|_1 \le 2 - s_{i,r}/(2k) \le 2 - s_{i+1,r}/k$$

when $s_{i+1,r} \le s_{i,r}/2$, that is, when $4(1 + \frac{1}{i+1})^4 \le k_r^{2^{-i}}$.

This proves the loop invariants (3) when $k_r^{2^{-i}} \ge \max\{(i+1)^2, 4(1 + \frac{1}{i+1})^4\}$, and that is $i \le i_r^*$ for some $i_r^* = \mathcal{O}(\log \log k_r)$. At this stage, the residual admits the decomposition $y^{(i_r^*,r)} + z^{(i_r^*,r)}$ with $|\operatorname{supp}(y^{(i_r^*,r)})| \le \mathcal{O}(\log^4 \log k)$ and $\|z^{(i,r)}\|_1 \le 2 - s_{i_r^*,r}/k$.

Now we change our choice of parameters and the loop invariants. In the i-th step ($i \ge i_r^* + 1$), we claim the following invariants hold at the beginning of the i-th step by changing $w_{i,r}$ to $w_{i,r} = 1$:

$$\begin{aligned} |\operatorname{supp}(\hat{y}^{(i,r)})| &\le s_{i,r} := (s_{i_r^*,r})^{2^{-(i-i_r^*-1)}} \\ \|\hat{z}^{(i,r)}\|_1 &\le 2 - s_{i,r}/k \end{aligned} \tag{4}$$

By our choice of i_r^* and the argument above the invariants hold when $i = i_r^* + 1$. Applying Lemma 12 with $w_{i,r} = 1$, we see that

$$|\operatorname{supp}(\hat{y}^{(i,r)})| \le \sqrt{s_{i,r}} \le s_{i+1,r}$$

and

$$\|\hat{z}^{(i,r)}\|_1 \le 2 - s_{i,r}/(2k_r) \le 2 - s_{i+1,r}/k,$$

whenever $s_{i,r} \geq 4$. This proves the loop invariants (4) when $s_{i,r} \geq \max\{4, \log\log k_r\}$, which holds when $i \leq i_r^* + i_r^+$ for some $i_r^+ = O(1)$ (recall that $s_{i_r^*} = O(\log^4 \log k_r)$). These steps leaves us a decomposition of the residual as $y^{(i,r)} + z^{(i,r)}$, where $|\operatorname{supp}(y^{(i,r)})| = s_{i,r} \leq \max\{4, \log\log k_r\}$ and $\|z^{(i,r)}\|_1 \leq 2 - s_{i,r}/k$.

Next, we start a new round by setting $k_{r+1} = s_{0,r+1} = s_{i_r^* + i_r^+ + 1, r}$. The loop invariants in (3) continue to hold in the base case $i = 0$. This completes proof of the claim that the loop invariants hold for $(0, r+1)$, provided that $k_{r+1} > 4$. Since $k_{r+1} \leq \log\log k_r$ and $k_0 = k$, the loop invariants in (3) hold for all $r \leq r_0$ for some $r_0 = O(\log^* k)$.

When $k_{r+1} \leq 4$, that is, there are at most 4 heavy hitter left, we shall recover all of them in one call to the weak system. Setting $w < 1/4$ in Lemma 12 yields that $|\operatorname{supp}(\hat{y})| < 1$; it thus must hold that $|\operatorname{supp}(\hat{y})| = 0$, or $\hat{y} = 0$, which means that all heavy hitters have recovered. This last call recovers $\hat{x}_{\text{final}}$, which has support size $O(1)$.

Support size of output. The support size of the output $\hat{x}$ is upper bounded by

$$|\operatorname{supp}(\hat{x}_{\text{final}})| + \sum_{r,i} |\operatorname{supp}(\hat{x}^{(i,r)})| \leq \mathcal{O}(1) + \sum_r \mathcal{O}(k_r) = \mathcal{O}(k).$$

Number of rows. In all rounds except the last round, the number of rows is bounded by

$$\mathcal{O}\left(\sum_{i=0}^{i_r^*} \frac{k^2}{(i+1)^2} \log n\right) + \mathcal{O}(i_r^+ \cdot k^2 \log n) = \mathcal{O}(k^2 \log n)$$

and the last round needs $\mathcal{O}(k^2 \log n)$ rows. The overall number of rows is therefore $m = \mathcal{O}(k^2 \log n \log^* k)$ as there are $\mathcal{O}(\log^* k)$ rounds.

Runtime. Each call to the weak system runs in $\mathcal{O}(k^6 \operatorname{poly}(\log n))$ time and there are $(\sum_r (i_r^* + i_r^+)) + 1 = \mathcal{O}(\log\log k \cdot \log^* k)$ calls. Each update of $y^{(i+1,r)} \leftarrow y^{(i,r)} - \Phi\hat{x}^{(i,r)}$ takes $\mathcal{O}(mk)$ since Φ has m rows and $|\operatorname{supp}(\hat{x})| = \mathcal{O}(k)$; there are $O(\log\log k \cdot \log^* k)$ such updates. The overall runtime is therefore $\mathcal{O}(k^6 \operatorname{poly}(\log n))$.

Storage of the sketching matrix. Each weak system uses $\mathcal{O}(k \log n)$ random $\mathcal{O}(k)$-wise independent hash function and needs space $\mathcal{O}(k^2 \log n)$ words. We have $\mathcal{O}(\log\log k \cdot \log^* k)$ such hash functions and thus the total storage for sketching matrix is $\mathcal{O}(k^2 \log n \cdot \log\log k \log^* k)$ words.

E Proof of Theorem 1

We first pick a matrix A using Theorem 3, setting the sparsity parameter to k^2 and $\epsilon = 1$. We also pick a matrix B satisfying the guarantees of Theorem 11, with sparsity $6k$, and a matrix C using Lemma 13 with sparsity parameter $6k$. Our sketching matrix Φ is the vertical concatenation of A, B and C. The total number of rows is $\mathcal{O}(k^2 \log n)$ for A and C, and $\mathcal{O}(k^2 \log n \log^* k)$ for B, for a total of $\mathcal{O}(k^2 \log n \log^* k)$ rows.

We first run the algorithm on Ax to obtain an $\mathcal{O}(k^2)$-sparse vector z such that $\|x - z\|_1 \leq 2\|x_{-k^2}\|_1$. Then we form $B(x-z)$ and using the query algorithm for B, we find an $\mathcal{O}(k)$-sparse vector w such that

$$\|(x - z) - w\|_\infty \leq \frac{1}{2k}\|x - z\|_1 \leq \frac{1}{k}\|x_{-k^2}\|_1. \tag{5}$$

Let $\mathcal{H}$ be the set of coordinates $i \in [n]$ such that $|x_i| > \frac{1}{k}\|x_{-k^2}\|_1$. We claim that $\mathcal{H} \subseteq \text{supp}(z) \cup \text{supp}(w)$; otherwise, it holds for $i \in \mathcal{H}$ that

$$|((x-z)-w)_i| = |x_i| > \frac{1}{k}\|x_{-k^2}\|_1,$$

which contradicts (5). The next step is to estimate x_i, for every $i \in \text{supp}(z) \cup \text{supp}(w)$, up to $(1/k)\|x_{-k^2}\|_1$ error. This argument is almost identical to [22], but we include it here for completeness. For every such i, define vector z' to be equal to z but with the i-th coordinate zeroed out. Then we run the point query algorithm of Lemma 13 on sparsity parameter $6k$ with sketch $C(x-z')$ to obtain a value $\hat{x}_i$ such that

$$|\hat{x}_i - x_i| = |\hat{x}_i - (x-z')_i| \leq \frac{1}{6k}\|(x-z')_{[n]\setminus\{i\}}\|_1 \leq \frac{1}{6k}\|x-z\|_1 \leq \frac{1}{3k}\|x_{-k^2}\|_1.$$

We note that $|\mathcal{H}| \leq k$ and, hence, by keeping the top $4k$ coordinates in magnitude, we shall include all elements in $\mathcal{H}$. Otherwise, there are at least $3k$ estimates of value at least $\frac{2}{3k}\|x_{-k^2}\|_1$ and so there are at least $3k$ coordinates of $x_{\text{supp}(z)\cup\text{supp}(w)}$ of magnitude at least $\frac{1}{3k}\|x_{-k^2}\|_1$, which is impossible. This concludes the proof of correctness.

Running time. The first step of obtaining z takes time $\mathcal{O}(k^3 \operatorname{poly}(\log n))$ by Theorem 3. The second step of obtaining w takes time $\mathcal{O}(k^6 \operatorname{poly}(\log n))$ by Theorem 11. The third step makes $\mathcal{O}(k^2)$ point queries. For each point query, it computes $C(x-z') = Cx - Cz'$, where Cx is part of the overall sketch and Cz' can be efficiently computed in $\mathcal{O}(k^4 \log n)$ time since C has $\mathcal{O}(k^2 \log n)$ rows and z' is $\mathcal{O}(k^2)$-sparse. Then the point query procedure itself runs in time $\mathcal{O}(k \log n)$ by Lemma 13. The total runtime of the third step is thus $\mathcal{O}(k^6 \log n)$. The overall runtime is dominated by that of the second step.

Storage space. The space to store A is $\mathcal{O}(k^2 \log n)$ words by Theorem 3. The space to store B is $\mathcal{O}(k^2 \log n \cdot \log^* k \cdot \log\log k)$ words by Theorem 11. The space to store C is $\mathcal{O}(k^2 \log n)$ words by taking C to be a fast Johnson-Lindenstrauss Transform matrix [15]. The overall storage space is dominated by that of B.

On Low-Risk Heavy Hitters and Sparse Recovery Schemes

Yi Li
Nanyang Technological University, Singapore
yili@ntu.edu.sg

Vasileios Nakos[1]
Harvard University, USA
vasileiosnakos@g.harvard.edu

David P. Woodruff[2]
Carnegie Mellon University, USA
dwoodruf@cs.cmu.edu

Abstract

We study the heavy hitters and related sparse recovery problems in the *low failure probability regime*. This regime is not well-understood, and the main previous work on this is by Gilbert et al. (ICALP'13). We recognize an error in their analysis, improve their results, and contribute new sparse recovery algorithms, as well as provide upper and lower bounds for the heavy hitters problem with low failure probability. Our results are summarized as follows:

1. (Heavy Hitters) We study three natural variants for finding heavy hitters in the strict turnstile model, where the variant depends on the quality of the desired output. For the weakest variant, we give a randomized algorithm improving the failure probability analysis of the ubiquitous Count-Min data structure. We also give a new lower bound for deterministic schemes, resolving a question about this variant posed in Question 4 in the IITK Workshop on Algorithms for Data Streams (2006). Under the strongest and well-studied ℓ_∞/ℓ_2 variant, we show that the classical Count-Sketch data structure is optimal for very low failure probabilities, which was previously unknown.
2. (Sparse Recovery Algorithms) For non-adaptive sparse-recovery, we give sublinear-time algorithms with low-failure probability, which improve upon Gilbert et al. (ICALP'13). In the adaptive case, we improve the failure probability from a constant by Indyk et al. (FOCS '11) to $e^{-k^{0.99}}$, where k is the sparsity parameter.
3. (Optimal Average-Case Sparse Recovery Bounds) We give matching upper and lower bounds in all parameters, including the failure probability, for the measurement complexity of the ℓ_2/ℓ_2 sparse recovery problem in the spiked-covariance model, completely settling its complexity in this model.

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

Keywords and phrases heavy hitters, sparse recovery, turnstile model, spike covariance model, lower bounds

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.19

Related Version Full version available at `https://arxiv.org/abs/1709.02919`.

[1] Supported in part by NSF grant IIS-1447471
[2] Supported in part by NSF grant CCF-1815840

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018).
Editors: Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer; Article No. 19; pp. 19:1–19:13

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

1 Introduction

Finding heavy hitters in data streams is one of the most practically and theoretically important problems in the streaming literature. Subroutines for heavy hitter problems, and in particular for the Count-Min sketch, are the basis for multiple problems on geometric data streams, including k-means and k-median clustering [14, 6, 2], as well as image acquisition [12]. Studying schemes for finding such heavy hitters has also led to important geometric insights in ℓ_p-spaces [1].

Abstractly, in the heavy hitters problem, we are asked to report all frequent items in a very long stream of elements coming from some universe. The main restriction is that the memory consumption should be much smaller than the universe size and the length of the stream. To rephrase the problem, consider a frequency vector $x \in \mathbb{R}^n$, where n is the size of the universe. Each element i in the data stream updates the frequency vector as $x_i \leftarrow x_i + 1$. At the end of the data stream, we wish to find the coordinates of x for which $|x_i| \geq \epsilon \|x\|_1$.

Oftentimes the problem is considered under a more general streaming model called the strict turnstile model, where arbitrary deletions and additions are allowed, but at all times the entries of x remain non-negative, that is $x_i \geq 0$. More formally, the frequency vector $x \in \mathbb{R}^n$ receives updates of the form (i, Δ), and each such update causes the change of x_i to $x_i + \Delta$, while ensuring that $x_i \geq 0$. The goal is to maintain a data structure such that upon query, the data structure returns the heavy hitters of the underlying vector x. The ℓ_p heavy hitters problem, for $p \geq 1$, then asks to find all coordinates i for which $x_i^p \geq \epsilon \|x\|_p^p$. The algorithm that treats the ℓ_1 case is the Count-min sketch [5], and the algorithm that treats the ℓ_2 case is the Count-Sketch [3]. Both algorithms are randomized and succeed with probability $1 - 1/\operatorname{poly}(n)$. In [5] the authors also suggest the "dyadic" trick for exchanging query time with space. Their "dyadic" trick allows for finding heavy hitters approximately in $\mathcal{O}(\frac{1}{\epsilon} \log^2 n)$ time, but with the downside of having a worse update time and a worse space consumption by an $\mathcal{O}(\log n)$ factor. The state of the art for heavy hitters is [17], where the authors give an algorithm that satisfies the ℓ_∞/l_p guarantee, has space $\mathcal{O}(\frac{1}{\epsilon} \log n)$, update time $\mathcal{O}(\log n)$, and query time $\mathcal{O}(\frac{1}{\epsilon} \operatorname{poly}(\log n))$. We note that the latter algorithm works in the more general setting of the turnstile model, where there is no constraint on x_i, in contrast to the strict turnstile model.

Another set of closely related problems occurs in the compressed sensing (CS) literature, which has seen broad applications to biomedical imaging, sensor networks, hand-held digital cameras, and monitoring systems, among other places. Sparse compressed sensing schemes were used for determining the attitudes[3], or 3-axis orientation, of spacecraft in [12].

Abstractly, in this problem we also seek to find the large coordinates of $x \in \mathbb{R}^n$ but with a different goal. Instead of finding all coordinates x_i for which $|x_i| \geq \epsilon \|x\|_1$, the CS problem seeks an approximation $\hat{x}$ to x such that the difference vector $x - \hat{x}$ has small norm. In particular, we consider the ℓ_2/ℓ_2 error metric, that is, we require that $\|x - \hat{x}\|_2^2 \leq (1+\epsilon)\|x_{-k}\|_2^2$, where x_{-k} is the vector x with the k largest entries (in magnitude) removed. If all ℓ_2 heavy hitters are found, it is clear that the norm of $x - \hat{x}$ can be made small, but the CS problem allows a small number of heavy hitters to be missed if their contribution to the approximation error $x - \hat{x}$ is small.

Gilbert et al. proposed the first sublinear-time algorithm for the ℓ_2/ℓ_2 problem that achieves $O(\frac{k}{\epsilon} \log \frac{n}{k})$ measurements with constant failure probability [9]. Earlier sublinear-time algorithms all contain several additional $\log n$ factors in their number of measurements.

[3] See `https://en.wikipedia.org/wiki/Attitude_control` for the notion of 'attitude' in this context.

The optimality of $O(\frac{k}{\epsilon}\log\frac{n}{k})$ measurements was shown by Price and Woodruff [21]. Later Gilbert et al. improved the failure probability to $n^{-k/\operatorname{poly}(\log k)}$ [10], while their number of measurements has a poor dependence on ϵ, which is at least ϵ^{-11}.

Despite the above works, our understanding of the complexity of heavy hitter and compressed sensing schemes on the error probability is very limited. The question regarding failure probability of these schemes is a natural one for two reasons: first, it is strongly connected with the existence of uniform schemes via the probabilistic method, and, second, being able to amplify the failure probability of an algorithm in a non-trivial way without making parallel repetitions of it, makes the algorithm much more powerful application-wise. For sparse recovery schemes, our goal is to obtain the same measurements but with smaller failure probability, something we find important both from a practical and theoretical perspective- obtaining the correct number of measurements in terms of all parameters ϵ, k, n, δ would be the end of the story for compressed sensing tasks, and a challenging quest; we note that previous algorithms achieved optimality with respect to ϵ, k, n only. From the practical side, a small enough failure probability would allow to re-use the same measurements all the time, since an application of the union-bound would suffice for all vectors that might appear in a lifetime; thus, application-wise, we could treat such a scheme as uniform. We start with formal definitions of the problems and then state in detail our improvements over previous work.

1.1 Problem Formulation

For $x \in \mathbb{R}^n$, we define $H_k(x)$ to be the set of its largest k coordinates in magnitude, breaking ties arbitrarily. For a set S let x_S be the vector obtained from x by zeroing out every coordinate $i \notin S$. We also define $x_{-k} = x_{[n]\setminus H_k(x)}$. For the ℓ_2/ℓ_2-sparse recovery results we define $H_{k,\epsilon}(x) = \{i \in [n] : |x_i|^2 \geq \frac{\epsilon}{k}\|x_{-k}\|_2^2\}$.

Two common models in the literature are the strict turnstile model and the (general) turnstile model.

Strict Turnstile Model: Both insertions and deletions are allowed, and it is guaranteed that at all times $x_i \geq 0$.

(General) Turnstile Model: Both insertions and deletions are allowed, but there is no guarantee about the value of x_i at any point in time.

A sketch $f : \mathbb{R}^m \to \mathbb{R}^n$ is a function that maps an n-dimensional vector to m dimensions. In this paper, all sketches will be linear, meaning $f(x) = Ax$ for some $A \in \mathbb{R}^{m\times n}$. The sketch length m will be referred to as the space of our algorithms.

▶ **Definition 1** (Heavy hitters). For $x \in \mathbb{R}^n$ and $p \geq 1$, a coordinate x_i is called an ϵ-heavy hitter in ℓ_p norm if $|x_i|^p \geq \epsilon\|x\|_p^p$. We consider the following three variants of the heavy hitters problem with different **guarantees**:

1. Return a list containing all ϵ-heavy hitters but no $\epsilon/2$-heavy hitters.
2. Return a list L of size $\mathcal{O}(1/\epsilon)$ containing all ϵ-heavy hitters along with estimates $\hat{x}_i$ such that $|x_i - \hat{x}_i|^p \leq \epsilon\|x_{-\lceil 1/\epsilon\rceil}\|_p^p$ for all $i \in L$.
3. Return a list of size $\mathcal{O}(1/\epsilon)$ containing all ϵ-heavy hitters.

When the algorithm is randomized, it has a parameter δ of failure probability; that is, the algorithm succeeds with probability at least $1 - \delta$.

The variant with Guarantee 2 above is also referred to as the ℓ_∞/ℓ_p problem. In this paper we focus on $p = 1$ and $p = 2$, with corresponding Count-Min [5] and Count-Sketch [3] data structures that have been studied extensively.

We note that the strongest guarantee is Guarantee 2. It is folklore that Guarantee 2 implies Guarantees 1 and 3, and Guarantee 1 clearly implies Guarantee 3. In applications, such as sparse recovery tasks, it is often the case that one does not need the full power of Guarantees 1 and 2, but rather is satisfied with Guarantee 3. The natural question that arises is whether one can gain some significant advantage under this Guarantee. Indeed, we show that Guarantee 3 allows the existence of a uniform scheme, i.e., one that works for all vectors, in the strict turnstile model with the same space, in contrast to the other two guarantees.

▶ **Definition 2** (ℓ_2/ℓ_2 sparse recovery)**.** An ℓ_2/ℓ_2-recovery system $\mathcal{A}$ consists of a distribution $\mathcal{D}$ on $\mathbb{R}^{m\times n}$ and a recovery algorithm $\mathcal{R}$. We will write $\mathcal{A} = (\mathcal{D}, \mathcal{R})$. We say that $\mathcal{A}$ satisfies the ℓ_2/ℓ_2 guarantee with parameters $(n, k, \epsilon, m, \delta)$ if for a signal $x \in \mathbb{R}^n$, the recovery algorithm outputs $\hat{x} = \mathcal{R}(\Phi, \Phi x)$ satisfying

$$\mathbb{P}_{\Phi\sim\mathcal{D}}\left\{\|\hat{x} - x\|_2^2 \le (1+\epsilon)\|x_{-k}\|_2^2\right\} \ge 1 - \delta.$$

In the above definition, each coordinate of Φx is called a *measurement* and the vector Φx is referred to as the measurement vector or just as the measurements. The probability parameter δ is referred to as the failure probability.

1.2 Our Results

Heavy hitters. Our first result is an improved analysis of the Count-Min sketch [5] for Guarantee 3 under the change of the hash functions from 2-wise to $\mathcal{O}(\frac{1}{\epsilon})$-wise independence. Previous analyses for Guarantees 1 and 2 use $\mathcal{O}(\frac{1}{\epsilon}\log\frac{n}{\delta})$ space; in contrast our analysis shows that this version of the Count-Min sketch satisfies Guarantee 3 with only $\mathcal{O}(\frac{1}{\epsilon}\log(\epsilon n) + \log(\frac{1}{\delta}))$ space. Notably, the $\frac{1}{\epsilon}$ factor does not multiply the $\log(\frac{1}{\delta})$ factor. This result has two important consequences. First, it gives a uniform scheme for Guarantee 3; second, it implies an improved analysis of the classic dyadic trick [5] for Guarantee 3 using $\mathcal{O}(\frac{1}{\epsilon}\log(\epsilon n) + \log n \log(\frac{\log \epsilon n}{\delta}))$ space. For constant δ, previous analyses of the dyadic trick needed space $\mathcal{O}(\frac{1}{\epsilon}\log n \log(\frac{\log n}{\epsilon}))$ but our analysis shows that $\mathcal{O}(\frac{1}{\epsilon}\log(\epsilon n) + \log(\epsilon n)\log\log(\epsilon n))$ space suffices. These results are summarized in Table 1.

Regarding the lower bound, we give the first bound for Guarantee 2 with $p = 2$, which is simultaneously optimal in terms of n, any $\epsilon > \frac{1}{n^{.99}}$, and the failure probability δ. That is, we prove that the space has to be $\Omega(\frac{1}{\epsilon}\log\frac{\epsilon n}{\delta})$, which matches the upper bound of Count-Sketch [3] whenever $\epsilon > \frac{1}{n^{.99}}$. A lower bound of $\Omega(\frac{1}{\epsilon}\log(\epsilon n))$ was given in [16] and is valid for the full range of parameters of ϵ and n, but previous analyses cannot be adapted to obtain non-trivial lower bounds for $\delta < \frac{1}{n}$. Indeed, the lower bound instances used in arguments in previous work have deterministic upper bounds using $O(\frac{1}{\epsilon}\log n)$ space.

We also show a new randomized lower bound of $\Omega(\frac{1}{\epsilon}(\log n + \sqrt{\log\frac{1}{\delta}}))$ space for $p = 1$, provided that $\frac{1}{\epsilon} > \sqrt{\log\frac{1}{\delta}}$. Although not necessarily optimal, this lower bound is the first to show that a term involving $\log\frac{1}{\delta}$ must multiply the $\frac{1}{\epsilon}$ factor for $p = 1$. The assumption that $\frac{1}{\epsilon} > \sqrt{\log\frac{1}{\delta}}$ is necessary, as there exist deterministic $O(\frac{1}{\epsilon^2})$ space algorithms for $p = 1$ [7, 20]. For deterministic algorithms satisfying Guarantee 3 with $p = 1$, we also show a lower bound of $\Omega(\frac{1}{\epsilon^2})$ measurements, which resolves Question 4 in the IITK Workshop on Algorithms for Data Streams [18].

Table 1 Summary of previous heavy hitter algorithms. The notation $\mathcal{O}(\cdot)$ for space complexity is suppressed, $\tilde{\mathcal{O}}(\cdot)$ hides logarithmic factors in n, $1/\epsilon$ and $1/\delta$.

Algorithm	Space	Guarantee	Query time
Count-Min [5]	$\frac{1}{\epsilon}\log n + \frac{1}{\epsilon}\log(\frac{1}{\delta})$	1, 2	$\tilde{\mathcal{O}}(n)$
This paper	$\frac{1}{\epsilon}\log(\epsilon n) + \log(\frac{1}{\delta})$	3	$\tilde{\mathcal{O}}(n)$
Dyadic Trick [5]	$\frac{1}{\epsilon}\log n \log(\frac{\log n}{\delta\cdot\epsilon})$	1, 2	$\tilde{\mathcal{O}}(\frac{1}{\epsilon})$
This paper	$\frac{1}{\epsilon}\log(\epsilon n) + \log(\epsilon n)\log(\frac{\log(\epsilon n)}{\delta})$	3	$\tilde{\mathcal{O}}(\frac{1}{\epsilon})$

Sparse Recovery. We summarize previous algorithms in Table 2. We give algorithms for the ℓ_2/ℓ_2 problem with failure probability much less than $1/\operatorname{poly}(n)$ whenever $k = \Omega(\log n)$. We present two novel algorithms, one running in $\mathcal{O}(k \operatorname{poly}(\log n))$ time and the other in $O(k^2 \operatorname{poly}(\log n))$ time with a trade-off in failure probability. Namely, the first algorithm has a larger failure probability than the second one. The algorithms follow a similar overall framework to each other but are instantiated with different parameters. We also show how to modify the algorithm of [10] to obtain an optimal dependence on ϵ, achieving a smaller failure probability along the way. All of these results are included in Table 2. Our algorithms, while constituting a significant improvement over previous work, are still not entirely optimal. We show, however, that at least in the spiked covariance model, which is a standard average-case model of input signals, we can obtain optimal upper and lower bounds in terms of the measurement complexity. Combined with the identification scheme from [10] we also obtain a scheme with decoding time nearly linear in k, assuming that $k = n^{\Omega(1)}$.

Besides the above non-adaptive schemes, we also make contributions, in terms of the failure probability, for adaptive schemes. For adaptive sparse recovery, Indyk et al. gave an algorithm under the ℓ_2/ℓ_2 guarantee [15] using $\mathcal{O}((k/\epsilon)\log(\epsilon n/k))$ measurements and achieving constant failure probability. In followup work [19] adaptive schemes were designed for other ℓ_p/ℓ_p error guarantees and improved bounds on the number of rounds were given; here our focus, as with the non-adaptive schemes we study, is on the error probability. We give a scheme that achieves failure probability $e^{-k^{1-\gamma}}$ for any constant γ, using the same number of measurements. Moreover, we present an algorithm for the regime when $k/\epsilon \leq \operatorname{poly}(\log n)$. Our scheme achieves the stronger ℓ_∞/ℓ_2 guarantee and fails with probability $1/\operatorname{poly}(\log n)$. Thus, our algorithms improve upon [15] in both regimes: in the high-sparsity regime we get an almost exponential improvement in k, and in the low-sparsity regime we get $1/\operatorname{poly}(\log n)$.

2 Our Techniques

Heavy hitters. All the schemes we give are for the strict turnstile model. Our first algorithm is based on a small but important tuning to the Count-Min sketch: we change the amount of independence in the hash functions from 2-wise to $\mathcal{O}(1/\epsilon)$-wise. Since the estimate of any coordinate is an overestimate of it, we are able to show that the set of $\mathcal{O}(1/\epsilon)$ coordinates with the largest estimates is a superset of the set of the ϵ-heavy hitters. Although changing the amount of independence might increase both the update and the query time by a multiplicative factor of $1/\epsilon$, we show that using fast multipoint evaluation of polynomials, we can suffer only a $\log^2(1/\epsilon)$ factor in the update time in the amortized case, and a $\log^2(1/\epsilon)$ factor in query time in the worst case. The above observation for the Count-Min sketch gives also an improvement on the well-known dyadic trick which appeared in the seminal paper of Cormode and Hadjieleftheriou [4].

Table 2 Summary of previous sparse recovery results and the results obtained in this paper. The notation $\mathcal{O}(\cdot)$ is suppressed. The paper [10] and the third result of our paper require $k = n^{\Omega(1)}$. The constants γ, α should be thought as arbitrarily small constants, say .001. We also note that in the regime $k/\epsilon \leq n^{1-\alpha}$, the decoding time of our first algorithm becomes $(k/\epsilon)\log^{2+\gamma} n$. The exponents in the poly$(\cdot)$ factors in [9] and [10] are at least 5, though the authors did not attempt at an optimization of these quantities. The exponent in the poly$(\cdot)$ factors in [17] is at least 3. We also note that our first result, in the regime $k/\epsilon \leq n^{1-\beta}$, gives also $k\log^{2+\gamma} n$ running time, thus improving the running time of [9].

Paper	Measurements	Decoding Time	Failure Probability
[3]	$\epsilon^{-1}k\log n$	$n\log n$	$1/\operatorname{poly}(n)$
[9]	$\epsilon^{-1}k\log(n/k)$	$\epsilon^{-1}k\operatorname{poly}(\log(n/k))$	$\Omega(1)$
[10]	$\epsilon^{-11}k\log(n/k)$	$k^2\cdot\operatorname{poly}(\epsilon^{-1}\log n)$	$(n/k)^{-k/\log^{13} k}$
[17]	$\epsilon^{-1}k\log n$	$\epsilon^{-1}k\cdot\operatorname{poly}(\log n)$	$1/\operatorname{poly}(n)$
This paper	$\epsilon^{-1}k\log(n/k)$	$\epsilon^{-1}k^{1+\alpha}\log^3 n$	$e^{-\sqrt{k}/\log^3 k}$
	$\epsilon^{-1}k\log(n/k)$	$\epsilon^{-1}k^2(\log k)\log^{2+\gamma}(n/k)$	$e^{-k/\log^3 k}$
	$\epsilon^{-1}k\log(\frac{n}{\epsilon k})$	$\epsilon^{-1}k^2\operatorname{poly}(\log n)$	$(n/k)^{-k/\log k}$

For the deterministic case, our improved analysis of the Count-Min sketch implies a deterministic algorithm that finds heavy hitters of all vectors $x \in \mathbb{R}^n_+$; moreover, we show how expanders that expand only on sets of size $\Theta(1/\epsilon)$ (or equivalently lossless condensers) lead to schemes in the strict turnstile model under Guarantee 3. Then we instantiate the Guruswami-Umans-Vadhan expander [13] properly to obtain an explicit algorithm. The idea of using expanders in the context of heavy hitters has been employed by Ganguly [8], although his result was for the ℓ_∞/ℓ_1 problem with $\Omega(1/\epsilon^2)$ space. Known constructions of these combinatorial objects are based on list decoding, and do not achieve optimal parameters. Any improvement on explicit constructions of these objects would immediately translate to an improved explicit heavy hitters scheme for the strict turnstile model.

Our deterministic lower bounds are based on choosing "bad input vectors" for the sketching matrix S based on several properties of S itself. Since the algorithm is deterministic, it must succeed even for these vectors.

Our randomized lower bounds come from designing a pair of distributions which must be distinguished by a heavy hitters algorithm with the appropriate guarantee. They are based on distinguishing a random Gaussian input from a random Gaussian input with a large coordinate planted in a uniformly random position. By Lipschitz concentration of the ℓ_1-norm and ℓ_2-norm in Gaussian space, we show that the norms in the two cases are concentrated, so we have a heavy hitter in one case but not the other. Typically, the planted large coordinate corresponds to a column in S of small norm, which makes it indistinguishable from the noise on remaining coordinates. The proof is carried out by a delicate analysis of the total variation distance of the distribution of the image of the input under the sketch in the two cases.

Non-Adaptive Sparse Recovery. Our result follows a similar framework to that of [9], though chooses more carefully the main primitives it uses and balances the parameters in a more effective way. Both schemes consist of $\mathcal{O}(\log k)$ so-called weak systems: a scheme that takes as input a vector x and returns a vector $\hat{x}$ which contains a 2/3 fraction of the heavy hitters of x (the elements with magnitude larger than $\frac{1}{\sqrt{k}}\|x_{-k}\|_2$) along with accurate estimates of (most of) them. Then it proceeds by considering the vector $x - \hat{x}$, which contains

at most $1/3$ of the heavy hitters of x. We then feed the vector $x - \hat{x}$ to the next weak-level system to obtain a new vector which contains at most $(2/3)(1/3)k = 2k/9$ of the heavy hitters. Proceeding in a similar fashion, after the i-th stage we will be left with at most $k/3^i$ heavy hitters.

Each weak system consists of an identification and an estimation part. The identification part finds a $2/3$ fraction of the heavy hitters while the estimation part estimates their values. For the identification part, the algorithm in [9] hashes n coordinates to $\Theta(k)$ buckets using a 2-wise independent hash function and then uses an error-correcting code in each bucket to find the heaviest element. Since, with constant probability, a heavy hitter will be isolated and its value will be larger than the 'noise' level in the bucket it is hashed to, it is possible to find a $2/3$ fraction of the heavy hitters with constant probability in $\mathcal{O}(k \operatorname{poly}(\log n))$ decoding time. Moreover, in each bucket we use a b-tree, which is a folklore data structure in the data stream literature, the special case of which ($b = 2$) first appeared in [5]. The estimation part is a different analysis of the folklore Count-Sketch data structure: we show that the estimation scheme can be implemented with an optimal dependence on ϵ, in contrast to the the expander-based scheme in [10], which gave a sub-optimal dependence on ϵ.

In this paper, we first design an algorithm with running time $\mathcal{O}(k^2 \operatorname{poly}(\log n))$, as in [10], and then improve it to $\mathcal{O}(k^{1+\alpha} \operatorname{poly}(\log n))$ time, but with a slightly larger failure probability. The key observation is that in the first round we find a constant fraction of the heavy hitters with e^{-k} failure probability, in the second round we find a constant fraction of the remaining heavy hitters with $e^{-k/2}$ failure probability, and so on, with polynomially decreasing number of measurements. In later rounds we can decrease the number of measurements at a slower rate so that the failure probability can be reduced by using more measurements while the optimality of the number of measurements is retained. Our suffering from the quadratic dependence on k in the runtime is due to the fact that our sensing matrix is very dense, with $\mathcal{O}(k)$ non-zeros per column. Hence, updating measurements $y \leftarrow y - \Phi\hat{x}$ will incur a running time proportional to $\|\hat{x}\|_0 \cdot k$, where $\hat{x}$ is a $\mathcal{O}(k)$-sparse vector.[4]

But, how do we achieve an almost linear time algorithm while beating the constant failure probability of [9]? The idea is to use again the same analysis, but without sharpening the failure probability in the first $(1/2)\log k$ steps. The first $(1/2)\log k$ rounds still fail with tiny probability, and once we reach round $(1/2)\log k$, we can afford to run the quadratic-time algorithm above, since our sparsity is now $\mathcal{O}(\sqrt{k})$. Hence we would expect the total algorithm to run in time $\mathcal{O}((\sqrt{k})^2 \operatorname{poly}(\log n)) = \mathcal{O}(k \operatorname{poly}(\log n))$. Putting everything together, we obtain substantial improvements over both [9] and [10].

A caveat of our approach, which is the reason we obtain $k^{1+\alpha}$ dependence on the decoding time, is the following. Since we do not want to store the whole matrix, our algorithms are implemented differently when $k \leq n^{1-\alpha/2}$ and $k \geq n^{1-\alpha/2}$. In the former case, we use $\log n = \Theta(\log(n/k))$ measurements per bucket in the identification step, in order to avoid inverting an $O(k)$-wise independent hash function. In the latter case, to compute the pre-image of an $O(k)$-wise independent hash function we just evaluate the hash function, which corresponds to a degree-$O(k)$ polynomial, in all places in time $O(n \log^2 k)$, and trivially

[4] We note an omission in the runtime analysis in [10]. Their measurement matrix contains $s = 2^i/i^c$ (where c is a constant) repetitions of an expander-based identification matrix (see Lemma 4.10 and Theorem 4.9 of [10]). Each repetition has at least one non-zero entry per column and thus the measurement matrix for the i-th iteration has at least s non-zero entries per column, which implies that when $i = \log k - 1$, each column has at least $\Omega(k/\operatorname{poly}(\log k))$ nonzero entries. Updating measurements $y \leftarrow y - \Phi\hat{x}$ will then take $\Omega(k^2/\operatorname{poly}(\log k))$ time, where $\hat{x}$ has $\Omega(k)$ nonzero coordinates. Therefore we would expect that the overall running time of the recovery algorithm will be $\tilde{\Omega}(k^2)$ instead of their claimed $\tilde{\mathcal{O}}(k^{1+\alpha})$.

find the pre-images. The asymptotic complexity of our algorithm in its full generality is dominated by the latter case, where we obtain an $\tilde{\mathcal{O}}(k^{1+\alpha})$ decoding time. We note that in the regime $k \leq n^{1-\alpha}$, the running time becomes $\mathcal{O}(k \log^{2+\gamma} n)$.

Adaptive Compressed Sensing. We start by implementing a $1/\text{poly}(\log n)$ failure probability version of the 1-sparse routine of [15]. We apply a preconditioning step before running [15] with a different setting of parameters; this preconditioning step gives us power for the next iteration, enabling us to achieve the desired failure probability in each round.

The lemma above leads to a scheme for ℓ_2/ℓ_2 in the low-sparsity regime, when $k < \text{poly}(\log n)$. The algorithm operates by hashing into $\text{poly}(\log n)$ buckets, determining the heavy buckets using a standard variant of Count-Sketch, and then running the 1-sparse recovery in each of these buckets. The improved algorithm for 1-sparse recovery is crucial here since it allows for a union bound over all buckets found.

For the general case of k-sparsity, we show that the main iterative loop of [15] can be modified so that it gives exponentially smaller failure probability in k. The idea is that, as more and more heavy hitters are found, it is affordable to use more measurements to reduce the failure probability. Interestingly and importantly for us, the failure probability per round is minimized in the first round, and in fact is increasing exponentially, although this was not exploited in [15]. Therefore, in the beginning we have exponentially small failure probability, but in later rounds we can use more measurements to boost the failure probability by making more repetitions. This part needs care in order not to blow up the number of measurements while achieving the best possible failure probability. We use a martingale argument to handle the dependency issue that arises from hashing coordinates into buckets, and thus avoid additional repetitions that would otherwise increase the number of of measurements.

The two algorithms above show how we can beat the failure probability of [15] for all values of k: we have $1/\text{poly}(\log n)$ for small k and $\exp(-k^{0.999})$ for large k, thus achieving asymptotic improvements in every case.

We note that although in the heavy hitters schemes we take into account the space to store the hash functions, in the sparse recovery we adopt the standard practice of not counting the space needed to store the measurement matrix, and therefore we use full randomness.

3 Formal Statement of Results

In this section we state all of our results and in subsequent sections we shall only give an outline of our improved analysis of Count-Min and our lower bound for Count-Sketch. The proofs of all other theorems can be found in the full version. The notations $\mathcal{O}_{a,b,\ldots}, \Omega_{a,b,\ldots}$ indicate that the constant in $\mathcal{O}$- and Ω-notations depend on $a, b, \ldots$.

3.1 Heavy Hitters

3.1.1 Upper Bounds

▶ **Theorem 3** (ℓ_1 Heavy Hitters Under Guarantee 3)**.** *There exists a data structure* DS *which finds the ℓ_1 heavy hitters of any $x \in \mathbb{R}^n$ in the strict turnstile model under Guarantee 3. In other words, we can sketch x, such that we can find a list L of $\mathcal{O}(k)$ coordinates that contains all ε-heavy hitters of x. The space usage is $\mathcal{O}(\frac{1}{\epsilon}\log(\epsilon n))$, the update time is amortized $\mathcal{O}(\log^2(\frac{1}{\epsilon})\log(\epsilon n))$ and the query time is $\mathcal{O}(n\log^2(\frac{1}{\epsilon})\log(\epsilon n))$.*

The following theorem follows by an improved analysis of the dyadic trick [4].

▶ **Theorem 4.** *There exists a data structure with space* $\mathcal{O}(\frac{1}{\epsilon}\log(\epsilon n)+\log(\epsilon n)\cdot\log(\frac{\log(\epsilon n)}{\delta}))$ *that finds the* ℓ_1 *heavy hitters of* $x\in\mathbb{R}^n$ *in the strict turnstile model under Guarantee 3 with probability at least* $1-\delta$*. The update time is* $\mathcal{O}(\log^2(\frac{1}{\epsilon})\log(\epsilon n)+\epsilon\log(\epsilon n)\log^2(\frac{1}{\epsilon})\log(\frac{\log(\epsilon n)}{\delta}))$ *amortized and the query time is* $\mathcal{O}(\frac{1}{\epsilon}(\log^2(\frac{1}{\epsilon})\log(\epsilon n)+\log(\epsilon n)\log^2(\frac{1}{\epsilon})\log(\frac{\log(\epsilon n)}{\delta}))))$.

▶ **Theorem 5** (Explicit ℓ_1 Heavy Hitters in the Strict Turnstile Model)**.** *There exists a fully explicit algorithm that finds the* ϵ*-heavy hitters of any vector* $x\in\mathbb{R}^n$ *using space* $\mathcal{O}(k^{1+\alpha}(\log(\frac{1}{\epsilon})\log n)^{2+2/\alpha})$*. The update time is* $\mathcal{O}(\mathrm{poly}(\log n))$ *and the query time is* $\mathcal{O}(n\cdot\mathrm{poly}(\log n))$*.*

3.1.2 Lower Bounds

▶ **Theorem 6** (Strict turnstile deterministic lower bound for Guarantees 1,2)**.** *Assume that* $n=\Omega(\epsilon^{-2})$*. Any sketching matrix* S *must have* $\Omega(\epsilon^{-2})$ *rows if, in the strict turnstile model, it is always possible to recover from* Sx *a set which contains all the* ϵ*-heavy hitters of* x *and contains no items which are not* $(\epsilon/2)$*-heavy hitters.*

▶ **Theorem 7** (Turnstile deterministic lower bound for Guarantee 3)**.** *Assume that* $n=\Omega(\epsilon^{-2})$*. Any sketching matrix* S *must have* $\Omega(\epsilon^{-2})$ *rows if, in the turnstile model, some algorithm never fails in returning a superset of size* $O(1/\epsilon)$ *containing the* ϵ*-heavy hitters. Note that it need not return approximations to the values of the items in the set which it returns.*

▶ **Theorem 8** (Randomized Turnstile ℓ_1-Heavy Hitters Lower Bound for Guarantees $1,2$)**.** *Assume that* $1/\epsilon\geq C\sqrt{\log(1/\delta)}$ *and suppose* $n\geq\left\lceil 64\epsilon^{-1}\sqrt{\log(1/\delta)}\right\rceil$*. Then for any sketching matrix* S*, it must have* $\Omega(\epsilon^{-1}\sqrt{\log(1/\delta)})$ *rows if, in the turnstile model, it succeeds with probability at least* $1-\delta$ *in returning a set containing all the* ϵ ℓ_1*-heavy hitters and containing no items which are not* $(\epsilon/2)$ ℓ_1*-heavy hitters.*

▶ **Theorem 9** (Randomized ℓ_2-Heavy Hitters Lower Bound with Guarantee 2)**.** *Suppose that* $\delta<\delta_0$ *and* $\epsilon<1/\epsilon_0$ *for sufficiently small absolute constants* $\delta_0,\epsilon_0\in(0,1)$ *and* $n\geq\left\lceil 64\epsilon^{-1}\log(6/\delta)\right\rceil$*. Then for any sketching matrix* S*, it must have* $\Omega(\epsilon^{-1}\log(1/\delta))$ *rows if it succeeds with probability at least* $1-\delta$ *in returning a set containing all the* ϵ*-heavy hitters and containing no items which are not* $(\epsilon/2)$*-heavy hitters.*

3.2 Non-Adaptive Sparse Recovery

▶ **Theorem 10.** *Let* $1\leq k\leq n$ *be integers and* $\gamma>0$ *be a constant. There exists an* ℓ_2/ℓ_2 *sparse recovery system* $\mathcal{A}=(\mathcal{D},\mathcal{R})$ *with parameters* $(n,k,\epsilon,\mathcal{O}_\gamma(k/\epsilon\log(n/\epsilon k)),\exp(-k/\log^3 k))$*. Moreover,* $\mathcal{R}$ *runs in time* $\mathcal{O}_\gamma(k^2\log^{2+\gamma}n)$*.*

In other words, there exists an ℓ_2/ℓ_2 *sparse recovery system that uses* $\mathcal{O}(k\epsilon\log(n/\epsilon k))$ *measurements, runs in time* $\mathcal{O}_\gamma(k^2\log^{1+\gamma}n)$*, and fails with probability* $\exp(-k/\log^3 k)$*.*

▶ **Theorem 11.** *Let* $1\leq k\leq n$ *be integers and* $\gamma>0$ *be a constant. There exists an* ℓ_2/ℓ_2 *sparse recovery system* $\mathcal{A}=(\mathcal{D},\mathcal{R})$ *with parameters* $(n,k,\epsilon,\mathcal{O}_\gamma(k/\epsilon\log(n/k)),\exp(-\sqrt{k}/\log^3 k))$*. Moreover,* $\mathcal{R}$ *runs in time* $\mathcal{O}_\gamma(k/\epsilon\log^{2+\gamma}n)$*. In other words, there exists an* ℓ_2/ℓ_2 *sparse recovery system that uses* $\mathcal{O}(k\epsilon\log(n/\epsilon k))$ *measurements, runs in time* $\mathcal{O}_\gamma(k\log^{2+\gamma}n)$*, and fails with probability* $\exp(-\sqrt{k}/\log^3 k)$*.*

▶ **Theorem 12.** *Suppose that* $k=n^{\Omega(1)}$*. There exists an* ℓ_2/ℓ_2 *sparse recovery system* $\mathcal{A}=(\mathcal{D},\mathcal{R})$ *with parameters* $\left(n,k,\epsilon,\mathcal{O}(\frac{k}{\epsilon}\log\frac{n}{\epsilon k}),(\frac{n}{k})^{-\frac{k}{\log k}}\right)$*. Moreover,* $\mathcal{R}$ *runs in* $\mathcal{O}(k^2/\epsilon\,\mathrm{poly}(\log n))$ *time. In other words, there exists an* ℓ_2/ℓ_2 *sparse recovery system that uses* $\mathcal{O}(k\epsilon\log(n/\epsilon k))$ *measurements, runs in time* $\mathcal{O}(k^2/\epsilon\,\mathrm{poly}(\log n))$*, and fails with probability* $(n/k)^{-k/\log k}$*.*

3.3 Adaptive Sparse Recovery

▶ **Theorem 13** (Entire regime of parameters). *Let $x \in \mathbb{R}^n$ and $\gamma > 0$ be a constant. There exists an algorithm that performs $\mathcal{O}((k/\epsilon)\log\log(\epsilon n/k))$ adaptive linear measurements on x in $\mathcal{O}(\log^* k \cdot \log\log(\epsilon n/k))$ rounds, and finds a vector $\hat{x} \in \mathbb{R}^n$ such that $\|x-\hat{x}\|_2^2 \le (1+\epsilon)\|x_{-k}\|_2^2$. The algorithm fails with probability at most $\exp(-k^{1-\gamma})$.*

▶ **Theorem 14** (low sparsity regime). *Let $x \in \mathbb{R}^n$ and parameters k, ϵ be such that $k/\epsilon \le c\log n$, for some absolute constant c. There exists an algorithm that performs $\mathcal{O}((k/\epsilon)\log\log n)$ adaptive linear measurements on x in $\mathcal{O}(\log\log n)$ rounds, and finds a vector $\hat{x} \in \mathbb{R}^n$ such that $\|x-\hat{x}\|_2 \le (1+\epsilon)\|x_{-k/\epsilon}\|_2$. The algorithm fails with probability at most $1/\operatorname{poly}(\log n)$.*

3.4 Spiked Covariance Model

In the spiked covariance model, the signal x is subject to the following distribution: we choose k coordinates uniformly at random, say, $i_1, \ldots, i_k$. First, we construct a vector $y \in \mathbb{R}^n$, in which each y_{k_i} is a uniform Bernoulli variable on $\{-\sqrt{\epsilon/k}, +\sqrt{\epsilon/k}\}$ and these k coordinate values are independent of each other. Then let $z \sim N(0, \frac{1}{n}I_n)$ and set $x = y + z$. We now present a non-adaptive algorithm (although the running time is slow) that uses $\mathcal{O}((k/\epsilon)\log(\epsilon n/k) + (1/\epsilon)\log(1/\delta))$ measurements and present a matching lower bound.

▶ **Theorem 15** (Upper Bound). *Assume that $(k/\epsilon)\log(1/\delta) \le \beta n$, where $\beta \in (0,1)$ is a constant. There exists an ℓ_2/ℓ_2 algorithm for the spiked-covariance model that uses $\mathcal{O}\left(\frac{k}{\epsilon}\log\frac{\epsilon n}{k} + \frac{1}{\epsilon}\log\frac{1}{\delta}\right)$ measurements and succeeds with probability $\ge 1-\delta$. Here the randomness is over both the signal and the algorithm.*

▶ **Theorem 16** (Lower Bound). *Suppose that $\delta < \delta_0$ for a sufficiently small absolute constant $\delta_0 \in (0,1)$ and $n \ge \lceil 64\epsilon^{-1}\log(6/\delta)\rceil$. Then any ℓ_2/ℓ_2-algorithm that solves with probability $\ge 1-\delta$ the ℓ_2/ℓ_2 problem in the spiked-covariance model must use $\Omega(\epsilon^{-1}\log(1/\delta))$ measurements.*

Combining with the lower bound from [21] (Section 4) we get a lower bound for the spiked covariance model of $\Omega((k/\epsilon)\log(n/k) + \log(1/\delta)/\epsilon)$. We note that although the lower bound is not stated for the spiked covariance model, inspection of the proof indicates that the hard instance is designed in that model.

4 ℓ_∞/ℓ_2 lower bound

This section is devoted to the proof of Theorem 9. The proof is based on designing a pair of hard distributions which cannot be distinguished by a small sketch. We show this by using rotational properties of the Gaussian distribution to reduce our problem to a univariate Gaussian mean estimation problem, which we show is hard to solve with low failure probability.

4.1 Toolkit

We list some facts in measure of concentration phenomenon in this subsection and omit their proofs owing to space limitations. Hereinafter we use $D_{TV}(\cdot,\cdot)$ to denote the total variation distance between two distributions.

▶ **Fact 1** (TVD Between Gaussians). $D_{TV}(N(0,I_r), N(\tau,I_r)) = \mathbb{P}_{g\sim N(0,1)}\{|g| \le \|\tau\|_2/2\}$.

▶ **Fact 2** (Concentration of ℓ_2-Norm). *Suppose $x \sim N(0,I_n)$ and $n \ge 18\ln(6/\delta)$. Then $\mathbb{P}\{\sqrt{n}/2 \le \|x\|_2 \le 3\sqrt{n}/2\} \ge 1-\delta/3$.*

▶ **Fact 3** (Univariate Tail Bound). *Let $g \sim N(0,1)$. There exists $\delta_0 > 0$ such that it holds for all $\delta < \delta_0$ that $\mathbb{P}\{|g| \leq 4\sqrt{\log(1/\delta)}\} \geq 1 - \delta/3$.*

In our proofs we are interested in lower bounding the number r of rows of a sketching matrix S.

4.2 Proof of the lower bound

Proof. Let the universe size be $n = \left\lceil 64\epsilon^{-1}\log(6/\delta)\right\rceil$, which is large enough in order for us to apply Fact 2 (note if the actual universe size is larger, we can set all but the first $\left\lceil 64\epsilon^{-1}\log(6/\delta)\right\rceil$ coordinates of our input to 0).

Let r be the number of rows of the sketch matrix S, where $r \leq n$. If $r > n$, then we immediately obtain an $\Omega(\epsilon^{-1}\log(1/\delta))$ lower bound. We can assume that S has orthonormal rows, since a change of basis to the row space of S can always be performed in a post-processing step.

Hard Distribution. Let I be a uniformly random index in $[n]$.

Case 1: Let η be the distribution $N(0, I_n)$, and suppose $x \sim \eta$. By Fact 2, $\|x\|_2 \geq \sqrt{n/2}$ with probability $1 - \delta/3$. By Fact 3, $|x_I| \leq 4\sqrt{\log(1/\delta)}$ with probability $1 - \delta/3$. Let $\mathcal{E}$ be the joint occurrence of these events, so that $\mathbb{P}(\mathcal{E}) \geq 1 - 2\delta/3$.

By our choice of n, it follows that if $\mathcal{E}$ occurs, then $x_I^2 \leq 16\log(1/\delta) \leq \frac{\epsilon}{2}\|x\|_2^2$, and therefore I cannot be output by an ℓ_2-heavy hitters algorithm.

Case 2: Let $y \sim N(0, I_n)$ and $x = \sqrt{\epsilon n}e_I + y$, where e_I denotes the standard basis vector in the I-th direction. By Fact 2, $\|y\|_2 \leq \frac{3\sqrt{n}}{2}$ with probability $1 - \delta/3$. By Fact 3, $|y_I| \leq 4\sqrt{\log(1/\delta)} < \sqrt{\epsilon n}/2$ with probability $1 - \delta/3$. Let $\mathcal{F}$ be the joint occurrence of these events, so that $\mathbb{P}(\mathcal{F}) \geq 1 - 2\delta/3$.

If event $\mathcal{F}$ occurs, then $|x_I| \geq 3\sqrt{\epsilon n} - 4\sqrt{\log(1/\delta)} \geq \frac{5\sqrt{\epsilon n}}{2}$, and so $x_I^2 \geq \frac{25\epsilon n}{4}$. We also have $\|x\|_2 \leq 3\sqrt{\epsilon n} + \frac{3\sqrt{n}}{2} \leq 2\sqrt{n}$, provided $\epsilon \leq 1/36$, and so $\|x\|_2^2 \leq 4n$. Consequently, $x_I^2 \geq \epsilon\|x\|_2^2$. Consequently, if $\mathcal{F}$ occurs, for an ℓ_2-heavy hitters algorithm to be correct, it must output I.

Conditioning. Let η' be the distribution of η conditioned on $\mathcal{E}$, and let γ' be the distribution of γ conditioned on $\mathcal{F}$. For a distribution μ on inputs y, we let $\bar{\mu}$ be the distribution of Sy.

Note that any ℓ_2-heavy hitters algorithm which succeeds with probability at least $1-\delta$ can decide, with probability at least $1 - \delta$, whether $x \sim \eta'$ or $x \sim \gamma'$. Hence, $D_{TV}(\bar{\eta'}, \bar{\gamma'}) \geq 1 - \delta$. Observe for any measurable set $A \subseteq \mathbb{R}^m$ it holds that

$$\left|\frac{\bar{\eta}(A) - \bar{\mu}(A)}{1 - \frac{2}{3}\delta} - (\bar{\eta}'(A) - \bar{\mu}'(A))\right| \leq \frac{2}{3}\delta,$$

and so it then follows that $D_{TV}(\bar{\eta}, \bar{\gamma}) \geq \left(D_{TV}(\bar{\eta'}, \bar{\gamma'}) - \frac{2\delta}{3}\right)\left(1 - \frac{2}{3}\delta\right) \geq 1 - \frac{7\delta}{3}$. Therefore, to obtain our lower bound, it suffices to show if the number r of rows of S is too small, then it cannot hold that $D_{TV}(\bar{\eta}, \bar{\gamma}) \geq 1 - 7\delta/3$.

Bounding the Total Variation Distance. Since S has orthonormal rows, by rotational invariance of the Gaussian distribution, the distribution of $\bar{\eta}$ is identical to $N(0, I_r)$ and the distribution of $\bar{\gamma}$ identical to $(3\sqrt{\epsilon n})S_I + N(0, I_r)$, where S_I is the I-th column of S.

Since S has orthonormal rows, by a Markov bound, for $9/10$ fraction of values of I, it holds that $\|S_I\|_2^2 \leq \frac{10r}{n}$. Call this set of columns T.

Let $\mathcal{G}$ be the event that $I \in T$, then $\mathbb{P}(\mathcal{G}) \geq 9/10$. It follows that

$$\begin{aligned} D_{TV}(\bar{\eta}, \bar{\gamma}) \leq \mathbb{P}(G) D_{TV}(\bar{\eta}, \bar{\gamma}|\mathcal{G}) + \mathbb{P}(\neg G) D_{TV}(\bar{\eta}, \bar{\gamma}|\neg\mathcal{G}) &\leq \mathbb{P}(G) D_{TV}(\bar{\eta}, \bar{\gamma}|\mathcal{G}) + 1 - \mathbb{P}(G) \\ &= 1 - \mathbb{P}(G)(1 - D_{TV}(\bar{\eta}, \bar{\gamma}|\mathcal{G})) \\ &\leq 1 - \frac{9}{10}(1 - D_{TV}(\bar{\eta}, \bar{\gamma}|\mathcal{G})). \end{aligned}$$

Hence, in order to deduce a contradiction that $D_{TV}(\bar{\eta}, \bar{\gamma}) < 1 - 7\delta/3$, it suffices to show that $D_{TV}(\bar{\eta}, \bar{\gamma}|\mathcal{G}) < 1 - 70\delta/27$.

The total variation distance between $N(0, I_r)$ and $(3\sqrt{\epsilon n})S_i + N(0, I_r)$ for a fixed $i \in T$ is, by rotational invariance and by rotating S_i to be in the same direction as the first standard basis vector e_1, the same as the total variation distance between $N(0, I_r)$ and $(3\sqrt{\epsilon n})\|S_i\|_2 e_1 + N(0, I_r)$, which is equal to the total variation distance between $N(0, 1)$ and $N(3\sqrt{\epsilon n}\|S_i\|_2, 1)$.

Using that $i \in T$ and so $\|S_i\|_2 \leq \sqrt{10r/n}$, we apply Fact 1 to obtain that the variation distance is at most $\mathbb{P}[|N(0,1)| \leq (3/2)\sqrt{\epsilon n} \cdot \sqrt{10r/n}]$. It follows that

$$D_{TV}(\bar{\eta}, \bar{\gamma} \mid \mathcal{G}) \leq \sum_{i \in T} \frac{1}{|T|} D_{TV}(\bar{\eta}, \bar{\gamma} \mid I = i) \leq \mathop{\mathbb{P}}_{g \sim N(0,1)} \left\{ |g| \leq \frac{3}{2}\sqrt{\epsilon n} \cdot \sqrt{\frac{10r}{n}} \right\},$$

and thus it suffices to show, when r is small, that $\mathbb{P}_{g \sim N(0,1)} \left\{ |g| \geq \frac{3}{2}\sqrt{10\epsilon r} \right\} > \frac{70\delta}{27}$. Observe that the left-hand is a decreasing function in r, and so it suffices to show the inequality above for $r = \alpha\epsilon^{-1}\log(1/\delta)$ for some $\alpha > 0$.

Invoking the well-known bound that (see, e.g., [11])

$$\mathop{\mathbb{P}}_{g \sim N(0,1)} \{g \geq t\} \geq \frac{1}{\sqrt{2\pi}} \cdot \frac{1}{2t} e^{-\frac{t^2}{2}}, \quad t \geq \sqrt{2},$$

we have that

$$\mathop{\mathbb{P}}_{g \sim N(0,1)} \left\{ |g| \geq \frac{3}{2}\sqrt{10\epsilon r} \right\} \geq \frac{3}{4\sqrt{5\pi}} \delta^{\frac{45}{4}\alpha} \frac{1}{\sqrt{\alpha \log(1/\delta)}} > \frac{70}{27}\delta$$

when α is small enough. Therefore it must hold that $r \geq \alpha\epsilon^{-1}\log(1/\delta)$ and the proof is complete. ◀

References

1 Zeyuan Allen Zhu, Rati Gelashvili, and Ilya P. Razenshteyn. Restricted isometry property for general p-norms. *IEEE Trans. Information Theory*, 62(10):5839–5854, 2016.

2 Vladimir Braverman, Gereon Frahling, Harry Lang, Christian Sohler, and Lin F. Yang. Clustering high dimensional dynamic data streams. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 576–585, 2017.

3 Moses Charikar, Kevin Chen, and Martin Farach-Colton. Finding frequent items in data streams. *Theoretical Computer Science*, 312(1):3–15, 2004.

4 Graham Cormode and Marios Hadjieleftheriou. Finding frequent items in data streams. *Proceedings of the VLDB Endowment*, 1(2):1530–1541, 2008.

5 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.

6 Gereon Frahling and Christian Sohler. Coresets in dynamic geometric data streams. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*, pages 209–217, 2005.

7 S. Ganguly and A. Majumder. CR-precis: A deterministic summary structure for update data streams. *eprint arXiv:cs/0609032*, 2006.

8 Sumit Ganguly. Data stream algorithms via expander graphs. In *International Symposium on Algorithms and Computation*, pages 52–63. Springer, 2008.

9 Anna C Gilbert, Yi Li, Ely Porat, and Martin J Strauss. Approximate sparse recovery: optimizing time and measurements. *SIAM Journal on Computing*, 41(2):436–453, 2012.

10 Anna C Gilbert, Hung Q Ngo, Ely Porat, Atri Rudra, and Martin J Strauss. ℓ_2/ℓ_2-foreach sparse recovery with low risk. In *International Colloquium on Automata, Languages, and Programming*, pages 461–472. Springer, 2013.

11 Robert D. Gordon. Values of mills' ratio of area to bounding ordinate and of the normal probability integral for large values of the argument. *Ann. Math. Statist.*, 12(3):364–366, 09 1941.

12 Rishi Gupta, Piotr Indyk, Eric Price, and Yaron Rachlin. Compressive sensing with local geometric features. *Int. J. Comput. Geometry Appl.*, 22(4):365, 2012.

13 Venkatesan Guruswami, Christopher Umans, and Salil Vadhan. Unbalanced expanders and randomness extractors from parvaresh–vardy codes. *Journal of the ACM (JACM)*, 56(4):20, 2009.

14 Piotr Indyk. Algorithms for dynamic geometric problems over data streams. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing, Chicago, IL, USA, June 13-16, 2004*, pages 373–380, 2004.

15 Piotr Indyk, Eric Price, and David P Woodruff. On the power of adaptivity in sparse recovery. In *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*, pages 285–294. IEEE, 2011.

16 Hossein Jowhari, Mert Sağlam, and Gábor Tardos. Tight bounds for lp samplers, finding duplicates in streams, and related problems. In *Proceedings of the thirtieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 49–58. ACM, 2011.

17 Kasper Green Larsen, Jelani Nelson, Huy L Nguyen, and Mikkel Thorup. Heavy hitters via cluster-preserving clustering. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 61–70. IEEE, 2016.

18 Andrew McGregor. Open problems in data streams and related topics: Iitk workshop on algorithms for data streams, 2006, 2007.

19 Vasileios Nakos, Xiaofei Shi, David P. Woodruff, and Hongyang Zhang. Improved algorithms for adaptive compressed sensing. In *ICALP*, 2018.

20 Jelani Nelson, Huy L. Nguyên, and David P. Woodruff. On deterministic sketching and streaming for sparse recovery and norm estimation. 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 627–638, 2012.

21 E. Price and D. P. Woodruff. $(1 + \epsilon)$-approximate sparse recovery. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, pages 295–304, Oct 2011.

Mildly Exponential Time Approximation Algorithms for Vertex Cover, Balanced Separator and Uniform Sparsest Cut

Pasin Manurangsi
University of California, Berkeley, USA
pasin@berkeley.edu

Luca Trevisan
University of California, Berkeley, USA
luca@berkeley.edu

Abstract

In this work, we study the trade-off between the running time of approximation algorithms and their approximation guarantees. By leveraging a structure of the "hard" instances of the Arora-Rao-Vazirani lemma [6, 42], we show that the Sum-of-Squares hierarchy can be adapted to provide "fast", but still exponential time, approximation algorithms for several problems in the regime where they are believed to be NP-hard. Specifically, our framework yields the following algorithms; here n denote the number of vertices of the graph and r can be any positive real number greater than 1 (possibly depending on n).

- A $\left(2 - \frac{1}{O(r)}\right)$-approximation algorithm for Vertex Cover that runs in $\exp\left(\frac{n}{2^{r^2}}\right) n^{O(1)}$ time.
- An $O(r)$-approximation algorithms for Uniform Sparsest Cut and Balanced Separator that runs in $\exp\left(\frac{n}{2^{r^2}}\right) n^{O(1)}$ time.

Our algorithm for Vertex Cover improves upon Bansal et al.'s algorithm [9] which achieves $\left(2 - \frac{1}{O(r)}\right)$-approximation in time $\exp\left(\frac{n}{r^r}\right) n^{O(1)}$. For Uniform Sparsest Cut and Balanced Separator, our algorithms improve upon $O(r)$-approximation $\exp\left(\frac{n}{2^r}\right) n^{O(1)}$-time algorithms that follow from a work of Charikar et al. [17].

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis

Keywords and phrases Approximation algorithms, Exponential-time algorithms, Vertex Cover, Sparsest Cut, Balanced Separator

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.20

Related Version A full version of the paper is available at `https://arxiv.org/abs/1807.09898`.

Funding This material is based upon work supported by the National Science under Grants No. CCF 1655215 and CCF 1815434.

1 Introduction

Approximation algorithms and fast (sub)exponential time exact algorithms are among the two most popular approaches employed to tackle NP-hard problems. While both have had their fair share of successes, they seem to hit roadblocks for a number of reasons; the PCP theorem [7, 5] and the theory of hardness of approximation developed from it have established, for many optimization problems, that trivial algorithms are the best one could hope for (in polynomial time). On the other hand, the Exponential Time Hypothesis (ETH) [30, 31] and

the fine-grained reductions surrounding it have demonstrated that "brute force" algorithms are, or at least close to, the fastest possible for numerous natural problems.

These barriers have led to studies in the cross-fertilization between the two fields, in which one attempts to apply both techniques simultaneously to overcome known lower bounds. Generally speaking, these works study the trade-offs between the running time of the algorithms and the approximation ratio. In other words, a typical question arising here is: what is the best running time for an algorithm with a given approximation ratio τ?

Optimization problems often admit natural "limited brute force" approximation algorithms that use brute force to find the optimal solution restricted to a subset of variables and then extend this to a whole solution. Similar to the study of fast exact algorithms for which a general motivating question is whether one can gain a noticeable speedup over "brute force", the analogous question when dealing with approximation algorithms is whether one can do significantly better than these limited brute force algorithms.

For example, let us consider the E3SAT problem, which is to determine whether a given 3CNF formula is satisfiable. The brute force (exact) algorithm runs in $2^{O(n)}$ time, while ETH asserts that it requires $2^{\Omega(n)}$ time to solve the problem. The optimization version of E3SAT is the Max E3SAT problem, where the goal is to find an assignment that satisfies as many clauses as possible. On the purely approximation front, a trivial algorithm that assigns every variable uniformly independently at random gives 7/8-approximation for Max E3SAT, while Hastad's seminal work [29] established NP-hardness for obtaining $(7/8+\varepsilon)$-approximation for any constant $\varepsilon > 0$. The "limited brute force" algorithm for Max E3SAT chooses a subset of $O(\epsilon n)$ variables, enumerates all possible assignments to those variables and picks values of the remaining variables randomly; this achieves $(7/8+\varepsilon)$-approximation in time $2^{O(\epsilon n)}$. Interestingly, it is known that running time of $2^{\Omega(\text{poly}(\varepsilon)n)}$ is necessary to gain a $(7/8+\varepsilon)$-approximation if one uses Sum-of-Squares relaxations [26, 53, 40], which gives some evidence that the running time of "limited brute force" $(7/8+\varepsilon)$ approximation algorithms for Max E3SAT are close to best possible.

In contrast to Max E3SAT, one can do much better than "limited brute force" for Unique Games. Specifically, Arora et al. [3] show that one can satisfy an ϵ fraction of clauses in a $(1-\epsilon)$-satisfiable instance of Unique Games in time $2^{n/exp(1/\epsilon)}$, a significant improvement over the trivial $2^{O(\varepsilon n)}$ time "limited brute force" algorithm. This algorithm was later improved by the celebrated algorithm of Arora, Barak and Steurer [2] that runs in time $2^{n^{poly(\epsilon)}}$.

A number of approximation problems, such as $(2-\varepsilon)$-approximation of Vertex Cover [38, 10], $(0.878\cdots+\varepsilon)$ approximation of Max Cut [35], and constant approximation of Non-uniform Sparsest Cut [18, 39] are known to be at least as hard as Unique Games, but are not known to be equivalent to Unique Games. If they were equivalent, the subexponential algorithm of [2] would also extend to these other problems. It is then natural to ask whether these problems admit subexponential time algorithms, or at least "better than brute force" algorithms. Indeed, attempts have been made to design such algorithms [2, 27], although these algorithms only achieve significant speed-up for specific classes of instances, not all worst case instances.

Recently, Bansal et al. [9] presented a "better than brute force" algorithm for Vertex Cover, which achieve a $(2-1/O(r))$-approximation in time $2^{O(n/r^r)}$. Note that the trade-off between approximation and running time is more analogous to the [3] algorithm for Unique Games than with the "limited brute force" algorithm for Max 3ESAT discussed above.

The algorithm of Bansal et al. is partially combinatorial and is based on a reduction to the Vertex Cover problem in bounded-degree graphs, for which better approximation algorithms are known compared to general graphs. Curiously, the work of Bansal et al. does

not subsume the best known polynomial time algorithm for vertex cover: Karakostas [32] shows that there is a polynomial time algorithm for vertex cover that achieves a $\left(2 - \frac{\Omega(1)}{\sqrt{\log n}}\right)$ approximation ratio, but if one set $r := \sqrt{\log n}$ in the algorithm of Bansal et al. one does not get a polynomial running time.

This overview raises a number of interesting questions: is it possible to replicate, or improve, the vertex cover approximation of Bansal et al. [9] using Sum-of-Square relaxations? A positive result would show that, in a precise sense, $(7/8 + \varepsilon)$ approximation of Max 3SAT is "harder" than $(2 - \varepsilon)$ approximation for Vertex Cover (since the former requires $\mathrm{poly}(\varepsilon) \cdot n$ rounds while the latter would be achievable with $n/\exp(1/\varepsilon)$ rounds). Is it possible to have a "better than brute force" approximation algorithm for vertex cover that recovers Karakostas's algorithm as a special case? Is it possible to do the same for other problems that are known to be Unique-Games-hard but not NP-hard, such as constant-factor approximation of balanced separator?

1.1 Our Results

In this work, we answer the above questions affirmatively by designing "fast" exponential time approximation algorithms for Vertex Cover, Uniform Sparsest Cut and (Uniform) Balanced Separator. For Vertex Cover, our algorithm gives $(2 - 1/O(r))$-approximation in time $\exp(n/2^{r^2})n^{O(1)}$ where n is the number of vertices in the input graph and r is a parameter that can be any real number at least one (and can depend on n). This improves upon the aforementioned recent algorithm of Bansal et al. [9] which, for a similar approximation ratio, runs in time $\exp(n/r^r)n^{O(1)}$. For the other two problems, our algorithms give $O(r)$-approximation in the same running time, which improves upon a known $O(r)$-approximation algorithms with running time $\exp(n/2^r)n^{O(1)}$ that follow from [17]:

▶ **Theorem 1** (Main Theorem). *For any $r > 1$, there are $\exp(n/2^{r^2})n^{O(1)}$-time $(2 - 1/O(r))$-approximation algorithm for Vertex Cover on n-vertex graphs, and, $\exp(n/2^{r^2})n^{O(1)}$-time $O(r)$-approximation algorithms for Uniform Sparsest Cut and Balanced Separator.*

We remark that, when we take $r = C\sqrt{\log n}$ for a sufficiently large constant C, our algorithms coincide with the best polynomial time algorithms known for all three problems [32, 6].

1.2 Other Related Works

To prove Theorem 1, we use the Sum-of-Square relaxations of the problems and employ the conditioning framework from [13, 52] together with the main structural lemma from Arora, Rao and Vazirani's work [6]. We will describe how these parts fit together in Section 2. Before we do so, let us briefly discuss some related works not yet mentioned.

Sum-of-Square Relaxation and the Conditioning Framework. The Sum-of-Square (SoS) algorithm [47, 49, 41] is a generic yet powerful meta-algorithm that can be utilized to any polynomial optimization problems. The approach has found numerous applications in both continuous and combinatorial optimization problems. Most relevant to our work is the conditioning framework developed in [13, 52]. Barak et al. [13] used it to provide an algorithm for Unique Games with similar guarantee to [2], while Raghavendra and Tan [52] used the technique to give improved approximation algorithms for CSPs with cardinality constraints. A high-level overview of this framework is given in Sections 2.2 and 2.3.

Approximability of Vertex Cover, Sparsest Cut and Balanced Separator. All three problems studied in our work are very well studied in the field of approximation algorithms and hardness of approximation. For Vertex Cover, the greedy 2-approximation algorithm has been known since the 70's (see e.g. [25]). Better $(2-\Omega(\frac{\log\log n}{\log n}))$-approximation algorithms were independently discovered in [11] and [45]. These were finally improved by Karakostas [32] who used the ARV Structural Theorem to provide a $(2-\Omega(1/\sqrt{\log n}))$-approximation for the problem. On the lower bound side, Hastad [29] show that $(7/6-\varepsilon)$-approximation for Vertex Cover is NP-hard. The ratio was improved in [21] to 1.36. The line of works that very recently obtained the proof of the (imperfect) 2-to-1 game conjecture [36, 19, 20, 37] also yield NP-hardness of $(\sqrt{2}-\varepsilon)$-approximate Vertex Cover as a byproduct. On the other hand, the Unique Games Conjecture (UGC) [33] implies that approximating Vertex Cover to within a factor $(2-\varepsilon)$ is NP-hard [38, 10]. We remark here that only Hastad reduction (together with Moshkovitz-Raz PCP [46]) implies an almost exponential lower bond in terms of the running time, assuming ETH. Putting it differently, it could be the case that Vertex Cover can be approximated to within a factor 1.5 in time say $2^{O(\sqrt{n})}$, without refuting any complexity conjectures or hypotheses mentioned here. Indeed, the question of whether a subexponential time $(2-\varepsilon)$-approximation algorithm for Vertex Cover exists for some constant $\varepsilon > 0$ was listed as an "interesting" open question in [2], and it remains so even after our work.

As for (Uniform) Sparsest Cut and Balanced Separator, they were both studied by Leighton and Rao who gave $O(\log n)$-approximation algorithms for the problems [43]. The ratio was improved in [6] to $O(\sqrt{\log n})$. In terms of hardness of approximation, these problems are *not* known to be NP-hard or even UGC-hard to approximate to even just 1.001 factor. (In contrast, the non-uniform versions of both problems are hard to approximate under UGC [18, 39].) Fortunately, inapproximability results of Sparsest Cut and Balanced Separator are known under stronger assumptions [22, 34, 51]. Specifically, Raghavendra et al. [51] shows that both problems are hard to approximate to any constant factor under the Small Set Expansion Hypothesis (SSEH) [50]. While it is not known whether SSEH follows from UGC, they are similar in many aspects, and indeed subexponential time algorithms for Unique Games [2, 13] also work for the Small Set Expansion problem. This means, for example, that there could be an $O(1)$-approximation algorithm for both problems in subexponential time without contradicting with any of the conjectures. Whether such algorithm exists remains an intriguing open question.

Fast Exponential Time Approximation Algorithms. As mentioned earlier, Bansal et al. [9] recently gave a "better than brute force" approximation algorithm for Vertex Cover. Their technique is to first observe that we can use branch-and-bound on the high-degree vertices; once only the low-degree vertices are left, they use Halperin's (polynomial time) approximation algorithm for Vertex Cover on bounded degree graphs [28] to obtain a good approximation. This approach is totally different than ours, and, given that the only way known to obtain $(2-\Omega(1/\sqrt{\log n}))$-approximation in polynomial time is via the ARV Theorem, it is unlikely that their approach can be improved to achieve similar trade-off as ours.

[9] is not the first work that gives exponential time approximation algorithms for Vertex Cover. Prior to their work, Bourgeois et al. [15] gives a $(2-1/O(r))$-approximation $\exp(n/r)$-time algorithm for Vertex Cover; this is indeed a certain variant of the "limited brute force" algorithm. Furthermore, Bansal et al. [9] remarked in their manuscript that Williams and Yu have also independently come up with algorithms with similar guarantees to theirs, but, to the best of our knowledge, Williams and Yu's work is not yet made publicly available.

For Sparsest Cut and Balanced Separator, it is possible to derive $O(r)$-approximation algorithms that run in $\exp(n/2^r)$-time from a work of Charikar et al. [17]. In particular, it

was shown in [17] that, for any metric space of n elements, if every subset of $n/2^r$ elements can be embedded isometrically into ℓ_1, then the whole space can be embedded into ℓ_1 with distortion $O(r)$. Since d-level of Sherali-Adams relaxations for both problems ensure that every d-size subset of the corresponding distance metric space can be embedded isometrically into ℓ_1, $(n/2^r)$-level of Sherali-Adams relaxations, which can be solved in $\exp(n/2^{\Omega(r)})$ time, give $O(r)$-approximation for both problems.

Organization. In the next section, we describe the overview of our algorithms. Then, in Section 3, we formalize the notations and state some preliminaries. The main lemma regarding conditioned SoS solution and its structure is stated in Section 4 (and proved in the appendix). This lemma is subsequently used in all our algorithms which are summarized in Section 5. We conclude our paper with several open questions in Section 6.

2 Overview of Technique

Our algorithms follow the "conditioning" framework developed in [13, 52]. In fact, our algorithms are very simple provided the tools from this line of work, and the ARV structural theorem from [6, 42]. To describe the ideas behind our algorithm, we will first briefly explain the ARV structural theorem and how conditioning works with Sum-of-Squares hierarchy in the next two subsections. Then, in the final subsection of this section, we describe the main insight behind our algorithms. For the ease of explaining the main ideas, we will sometimes be informal in this section; all algorithms and proofs will be formalized in the sequel.

Due to space constraint, we will focus only on the c-Balanced Separator problem; the algorithms for the two other problems can be found in the full version. In the c-Balanced Separator problem, we are given a graph $G = (V, E)$ and the goal is to find a partition of V into S_0 and $S_1 = V \setminus S_0$ that minimizes the number of edges across the cut (S_0, S_1) while also ensuring that $|S_0|, |S_1| \geqslant c'n$ for some constant $c' \in (0, c)$ where $n = |V|$. Note that the approximation ratio is the ratio between the number of edges cut by the solution and the optimal under the condition $|S_0|, |S_1| \geqslant cn$. (That is, this is a pseudo approximation rather than a true approximation; this is also the notion used in previous works [43, 6].) For the purpose of exposition, we focus only on the case where $c = 1/3$.

2.1 The ARV Structural Theorem

The geometric relaxation used in [6] embeds each vertex $i \in V$ into a point $v_i \in \mathbb{R}^d$ such that $\|v_i\|_2 = 1$. For a partition (S_0, S_1), the intended solution is $v_i = v_\emptyset$ if $i \in S_0$ and $v_i = -v_\emptyset$ otherwise, where $v_\emptyset$ is some unit vector. As a result, the objective function here is $\sum_{(i,j)\in E} \frac{1}{4}\|v_i - v_j\|_2^2$, and the cardinality condition $|S_0|, |S_1| \geqslant n/3$ is enforced by $\sum_{i,j\in V} \|v_i - v_j\|_2^2 \geqslant 8n/9$. Furthermore, Arora et al. [6] also employ the triangle inequality: $\|v_i - v_j\|_2^2 \leqslant \|v_i - v_k\|_2^2 + \|v_k - v_j\|_2^2$ for all $i, j, k \in V$. In other words, this relaxation can be written as follows.

$$\text{minimize} \quad \sum_{(i,j)\in E} \frac{1}{4}\|v_i - v_j\|_2^2 \tag{1}$$

$$\text{subject to} \quad \sum_{i,j\in[n]} \|v_i - v_j\|_2^2 \geqslant 8n/9 \tag{2}$$

$$\|v_i\|_2^2 = 1 \qquad \forall i \in V \tag{3}$$

$$\|v_i - v_j\|_2^2 \leqslant \|v_i - v_k\|_2^2 + \|v_k - v_j\|_2^2 \qquad \forall i, j, k \in V \tag{4}$$

Note here that the above relaxation can be phrased as a semidefinite program and hence can be solved to arbitrarily accuracy in polynomial time. The key insight shown by Arora et al. is that, given a solution $\{v_i\}_{i\in V}$ to the above problem, one can find two sets of vertices T, T' that are $\Omega(1/\sqrt{\log n})$ apart from each other, as stated below. Note that this version is in fact from [42]; the original theorem of [6] has a worst parameter with $\Delta = \Omega((\log n)^{2/3})$.

▶ **Theorem 2** (ARV Structural Theorem [6, 42]). *Let $\{v_i\}_{i\in V}$ be any vectors in $\mathbb{R}^d$ satisfying* (2), (3), (4). *There exist $T, T' \subseteq V$ each of size $\Omega(n)$ such that, for every $i \in T$ and $j \in T'$, $\|v_i - v_j\|_2^2 \geqslant \Delta = \Omega(1/\sqrt{\log n})$. Moreover, such sets can be found in randomized polynomial time.*

Note that, given the above theorem, it is easy to arrive at the $\Omega(1/\sqrt{\log n})$-approximation algorithm for balanced separator. In particular, we can pick a number θ uniformly at random from $[0, \Delta)$ and then output $S_0 = \{i \in V \mid \exists j \in T, \|v_i - v_j\|_2^2 \leqslant \theta\}$ and $S_1 = V \setminus S_0$. It is easy to check that the probability that each edge $(i, j) \in E$ is cut is at most $\|v_i - v_j\|_2^2/\Delta = O(\sqrt{\log n} \cdot \|v_i - v_j\|_2^2)$. Moreover, we have $|S_0| \geqslant |T| \geqslant \Omega(n)$ and $|S_1| \geqslant |T'| \geqslant \Omega(n)$, meaning that we have arrived at an $O(\sqrt{\log n})$-approximate solution for Balanced Separator.

An interesting aspect of the proof of [42] is that the bound on Δ can be improved if the solution $\{v_i\}_{i\in V}$ is "hollow" in the following sense: for every $i \in V$, the ball of radius[1] 0.1 around i contains few other vectors v_j's. In particular, if there are only m such v_j's, then Δ can be made $\Omega(1/\sqrt{\log m})$, instead of $\Omega(1/\sqrt{\log n})$ in the above version. We will indeed use this more fine-grained version in our algorithms. To the best of our knowledge, this version of the theorem has not yet been used in other applications of the ARV Theorem.

▶ **Theorem 3** (Refined ARV Structural Theorem [6, 42]). *Let $\{v_i\}_{i\in V}$ be any vectors in $\mathbb{R}^d$ satisfying* (2), (3), (4) *and let $m = \max_{i\in V} |\{j \in V \mid \|v_i - v_j\|_2^2 \leqslant 0.01\}|$. There exist $T, T' \subseteq V$ each of size $\Omega(n)$ such that, for every $i \in T$ and $j \in T'$, $\|v_i - v_j\|_2^2 \geqslant \Delta = \Omega(1/\sqrt{\log m})$. Moreover, such sets can be found in randomized polynomial time.*

2.2 Conditioning in Sum-of-Square Hierarchies

Another crucial tool used in our algorithm is Sum-of-Square hierarchy and the conditioning technique developed in [13, 52]. Perhaps the most natural interpretation of the sum-of-square solution with respect to the conditioning operation is to view the solution as local distributions. One can think of a degree-d sum-of-square solution for Balanced Separator as a collection of local distributions μ_S over $\{0,1\}^S$ for subsets of vertices $S \subseteq V$ of sizes at most d that satisfies certain consistency and positive semi-definiteness conditions, and additional linear constraints corresponding to $|S_0|, |S_1| \geqslant n/3$ and the triangle inequalities. More specifically, for every $U \subseteq V$ and every $\phi : U \to \{0,1\}$, the degree-d sum-of-squares solution gives us $\Pr_{\mu_U}[\forall j \in U, j \in S_{\phi(j)}]$ which is a number between zero and one. The consistency constraints ensure that these distributions are locally consistent; that is, for every $U' \subseteq U \in \{0,1\}$, the marginal distribution of μ_U on U' is equal to $\mu_{U'}$. We remark here that, for Balanced Separator and other problems considered in this work, a solution to the degree-d SoS relaxation for them can be found in time $\binom{n}{d}^{O(1)} = O(n/d)^{O(d)}$.

The consistency constraints on these local distributions allow us to define conditioning on local distributions in the same ways as typical conditional distributions. For instance, we can condition on the event $i \in S_0$ if $\Pr_{\mu_i}[i \in S_0] \neq 0$; this results in local distributions

[1] Here 0.1 can be changed to arbitrary positive constant.

$\{\mu'_U\}_{U\subseteq V,|U|\leqslant d-1}$ where μ'_U is the conditional distribution of $\mu_{U\cup\{i\}}$ on the event $i \in S_0$. In other words, for all $\phi : U \to \{0,1\}$,

$$\Pr_{\mu'_U}[\forall j \in U, j \in S_{\phi(j)}] = \frac{\Pr_{\mu_{U\cup\{i\}}}\left[i \in S_0 \wedge \left(\forall j \in U, j \in S_{\phi(j)}\right)\right]}{\Pr_{\mu_i}[i \in S_0]}.$$

Notice that the local distributions are now on subsets of at most $d-1$ vertices instead d vertices as before. That is, the conditioned solution is a degree-$(d-1)$ solution.

As for the semi-definiteness constraint, it suffices for the purpose of this discussion to think about only the degree-2 solution case. For this case, the semi-definiteness constraint in fact yields unit vectors $v_\emptyset, \{v_j\}_{j\in V}$ such that

$$\Pr_{\mu_i}[i \in S_0] = \frac{1 + \langle v_\emptyset, v_i\rangle}{2} \qquad \forall i \in V,$$

$$\Pr_{\mu_{\{i,j\}}}[i, j \in S_0] = \frac{1 + \langle v_\emptyset, v_i\rangle + \langle v_\emptyset, v_j\rangle + \langle v_i, v_j\rangle}{4} \qquad \forall i, j \in V.$$

It is useful to also note that the probability that i, j are on different side of the cut is exactly equal to $\frac{1}{4}\|v_i - v_j\|_2^2$; this is just because

$$\begin{aligned}\Pr_{\mu_{\{i,j\}}}[Y_i \neq Y_j] &= \Pr_{\mu_i}[i \in S_0] + \Pr_{\mu_j}[j \in S_0] - 2\Pr_{\mu_{\{i,j\}}}[i \in S_0 \wedge j \in S_0] \\ &= \frac{1 - \langle v_i, v_j\rangle}{2} = \frac{1}{4}\|v_i - v_j\|_2^2, \end{aligned} \tag{5}$$

where Y_i, Y_j are boolean random variables such that $i \in S_{Y_i}$ and $j \in S_{Y_j}$.

Finally, note that the constraints for $|S_0|, |S_1| \geqslant n/3$ and the triangle inequalities are those that, when written in vector forms, translate to (2) and (4) from the ARV relaxation.

2.3 Our Algorithms: Combining Conditioning and ARV Theorem

The conditioning framework initiated in [13, 52] (and subsequently used in [8, 54, 44]) typically proceeds as follows: solve for a solution to a degree-d Sum-of-Square relaxation of the problem for a carefully chosen value of d, use (less than d) conditionings to make a solution into an "easy-to-round" degree-$O(1)$ solution, and finally round such a solution.

To try to apply this with the Balanced Separator problem, we first have to understand what are the "easy-to-round" solutions for the ARV relaxation. In this regards, first observe that, due to the more refined version of the ARV Theorem (Theorem 3), the approximation ratio is actually $O(\sqrt{\log m})$ which can be much better than $O(\sqrt{\log n})$. In particular, if $m \leqslant 2^{O(r^2)}$, this already yields the desired $O(r)$-approximation algorithm. This will be one of the "easy-to-round" situations. Observe also that we can in fact relax the requirement even further: it suffices if $|\{j \in V \mid \|v_i - v_j\|_2^2 \leqslant 0.01\}| \leqslant m$ holds for a constant fraction of vertices $i \in V$. This is because we can apply Theorem 3 on only the set of such i's which would still result in well-separated set of size $\Omega(n)$. Recall also that from (5) the condition $\|v_i - v_j\|_2^2 \leqslant 0.01$ is equivalent to $\Pr_{\mu_{\{i,j\}}}[Y_i \neq Y_j] \leqslant 0.04$.

Another type of easy-to-round situation is when, for most (i.e. $0.9n$) of $i \in V$, $\Pr_{\mu_i}[i \in S_0] \notin [0.2, 0.8]$. In this case, $T = \{i \in V \mid \Pr_{\mu_i}[i \in S_0] < 0.2\}$ and $T' = \{j \in V \mid \Pr_{\mu_j}[j \in S_0] > 0.8\}$ is a pair of large well-separated sets; it is not hard to argue that both T, T' are at least $\Omega(n)$ and that, for every $i \in T$ and $j \in T'$, $\|v_i - v_j\|_2^2$ is at least 0.6.

To recap, it suffices for us to condition degree-d solution so that we end up in one of the following two "easy-to-round" cases to get an $O(r)$-approximation for the problem.

1. For at least $n/100$ vertices $i \in V$, we have $|\{j \in V \mid \Pr_{\mu_{\{i,j\}}}[Y_i \neq Y_j] \leqslant 0.04\}| \leqslant 2^{O(r^2)}$.
2. For at least $9n/10$ vertices $i \in V$, we have $\Pr_{\mu_i}[i \in S_0] \notin [0.2, 0.8]$.

Here we will pick our d to be $n/2^{r^2}$; the running time needed to solve for such a solution is indeed $O(n/d)^{O(d)} = \exp(n/2^{\Omega(r^2)})n^{O(1)}$ as claimed. Now, suppose that we have a degree-d solution that does not belong to any of the two easy-to-round cases as stated above. This means that there must be $i \in V$ such that $\Pr_{\mu_i}[i \in S_0] \notin [0.2, 0.8]$ and that $|\{j \in V \mid \Pr_{\mu_{\{i,j\}}}[Y_i \neq Y_j] \leqslant 0.04\}| > 2^{O(r^2)}$. For simplicity, let us also assume for now that $\Pr_{\mu_i}[i \in S_0] = 0.5$. We will condition on the event $i \in S_0$; let the local distributions after conditioning be $\{\mu'_U\}_{U \subseteq V, |U| \leqslant d-1}$. Consider each $j \in V$ such that $\Pr_{\mu_{\{i,j\}}}[Y_i \neq Y_j] \leqslant 0.04$. Observe first that, before the conditioning, we have $\Pr_{\mu_j}[j \in S_0] \geqslant \Pr_{\mu_i}[i \in S_0] - \Pr_{\mu_{\{i,j\}}}[Y_i \neq Y_j] > 0.4$ and $\Pr_{\mu_j}[j \in S_0] \geqslant \Pr_{\mu_i}[i \in S_0] + \Pr_{\mu_{\{i,j\}}}[Y_i \neq Y_j] < 0.6$.

On the other hand, after the conditioning, we have

$$\Pr_{\mu'_j}[j \in S_0] = \frac{\Pr_{\mu_{\{i,j\}}}[i \in S_0, j \in S_0]}{\Pr_{\mu_i}[i \in S_0]} \geqslant \frac{\Pr_{\mu_i}[i \in S_0] - \Pr_{\mu_{\{i,j\}}}[Y_i \neq Y_j]}{\Pr_{\mu_i}[i \in S_0]} > 0.9.$$

Thus, this conditioning makes at least 2^{r^2} vertices j's such that $\Pr_{\mu_j}[j \in S_0] \in [0.2, 0.8]$ beforehand satisfy $\Pr_{\mu'_j}[j \in S_0] \notin [0.2, 0.8]$ afterwards. If we ignore how conditioning affects the remaining variables for now, this means that, after $n/2^{r^2}$ such conditioning all vertices $j \in V$ must have $\Pr_{\mu_j}[j \in S_0] \in [0.2, 0.8]$. Hence, we have arrived at an "easy-to-round" solution and we are done! The effect to the other variables that we ignored can easily be taken into account via a simple potential function argument and by considering conditioning on both $i \in S_0$ and $i \in S_1$; this part of the argument can be found in Section 4. This concludes the overview of our algorithm.

3 Preliminaries

3.1 Sum-of-Square Hierarchy and Conditioning

We define several notations regarding the Sum-of-Square (SoS) Hierarchy; these notations are based mainly on [12, 48]. We will only state preliminaries necessary for our algorithms. We recommend interested readers to refer to [48, 14] for a more thorough survey on SoS.

Let $\mathbb{R}_d[X_1, \ldots, X_n]$ denote the set of all polynomials on $X_1, \ldots, X_n$ of total degree at most d. First, we define *pseudo-expectation*, which represents solutions to SoS Hierarchy:

▶ **Definition 4** (Pseudo-Expectation)**.** A *degree-d pseudo-expectation (p.e.)* is a linear operator $\tilde{\mathbb{E}} : \mathbb{R}_d[X_1, \ldots, X_n] \to \mathbb{R}$ that satisfies the following:

- (Normalization) $\tilde{\mathbb{E}}[1] = 1$.
- (Linearity) For any $p \in \mathbb{R}_d[X_1, \ldots, X_n]$ and $q \in \mathbb{R}_d[X_1, \ldots, X_n]$, $\tilde{\mathbb{E}}[p + q] = \tilde{\mathbb{E}}[p] + \tilde{\mathbb{E}}[q]$.
- (Positivity) For any $p \in \mathbb{R}_{\lfloor d/2 \rfloor}[X_1, \ldots, X_n]$, $\tilde{\mathbb{E}}[p^2] \geqslant 0$.

Furthermore, $\tilde{\mathbb{E}}$ is said to be *boolean* if $\tilde{\mathbb{E}}[(X_i^2 - 1)p] = 0$ for all $p \in \mathbb{R}_{d-2}[X_1, \ldots, X_n]$.

Observe that, while $\tilde{\mathbb{E}}$ is a function over infinite domain, $\tilde{\mathbb{E}}$ has a succinct representation: due to its linearity, it suffices to specify the values of all monomials of total degree at most d and there are $n^{O(d)}$ such monomials. Furthermore, for boolean $\tilde{\mathbb{E}}$, we can save even further since it suffices to specify only products of at most d different variables. There are only $O(n/d)^{O(d)}$ such terms. From now on, we will only consider boolean pseudo-expectations. Note also that we use X_i as ± 1 variables instead of $0, 1$ variable as used in the proof overview. (Specifically, in the language of the overview section, $\Pr_{\mu_i}[i \in S_0]$ is now equal to $\tilde{\mathbb{E}}[(1 - X_i)/2]$.)

▶ **Definition 5.** A *system of polynomial constraints* $(\mathcal{P}, \mathcal{Q})$ consists of a set of equality constraints $\mathcal{P} = \{p_i = 0\}_{i \in |\mathcal{P}|}$ and a set of inequality constraints $\mathcal{Q} = \{q_j \geqslant 0\}_{j \in |\mathcal{Q}|}$, where p_i, q_j are polynomials over $X_1, \ldots, X_n$.

For every $S \subseteq [n]$, we use X_S to denote the monomial $\prod_{i \in S} X_i$. Furthermore, for every $S \subseteq [n]$ and every $\phi : S \to \{\pm 1\}$, let X_ϕ be the polynomial $\prod_{i \in S} (1 + \phi(i) X_i)$. A boolean degree-$d$ p.e. $\tilde{\mathbb{E}}$ is said to *satisfy* a system $(\mathcal{P}, \mathcal{Q})$ if the following conditions hold:

- For all $p \in \mathcal{P}$ and all $S \subseteq [n]$ such that $|S| \leqslant d - \deg(p)$, we have $\tilde{\mathbb{E}}[X_S p] = 0$.
- For all $q \in \mathcal{Q}$, all $S \subseteq [n]$ s.t. $|S| \leqslant d - \deg(q)$ and all $\phi : S \to \{\pm 1\}$, we have $\tilde{\mathbb{E}}[X_\phi q] \geqslant 0$.

It is not hard to verify that finding a degree-d p.e. satisfying $(\mathcal{P}, \mathcal{Q})$ can be written as a semidefinite program of size $O(n/d)^{O(d)} \cdot |\mathcal{P}| \cdot |\mathcal{Q}|$ and hence can be solved in $\mathrm{poly}(O(n/d)^d \cdot |\mathcal{P}| \cdot |\mathcal{Q}|)$ time. Finally, we define conditioning in terms of pseudo-expectation:

▶ **Definition 6** (Conditioning)**.** Let $\tilde{\mathbb{E}} : \mathbb{R}_d[X_1, \ldots, X_n] \to \mathbb{R}$ be any boolean degree-d pseudo-expectation for some $d > 2$. For any $b \in \{\pm 1\}$ such that $\tilde{\mathbb{E}}[X_i] \neq -b$, we denote the conditional pseudo-expectation of $\tilde{\mathbb{E}}$ on $X_i = b$ by $\tilde{\mathbb{E}}|_{X_i = b} : \mathbb{R}_{d-1}[X_1, \ldots, X_n] \to \mathbb{R}$ where $\tilde{\mathbb{E}}|_{X_i = b}[p] = \frac{\tilde{\mathbb{E}}[p(1 + bX_i)]}{\tilde{\mathbb{E}}[1 + bX_i]}$ for all $p \in \mathbb{R}_{d-1}[X_1, \ldots, X_n]$.

The proposition below is simple to check, using the identity $(1 + bX_i) = \frac{1}{2}(1 + bX_i)^2$.

▶ **Proposition 7.** *If $\tilde{\mathbb{E}}$ satisfies a system $(\mathcal{P}, \mathcal{Q})$, then $\tilde{\mathbb{E}}|_{X_i = b}$ also satisfies $(\mathcal{P}, \mathcal{Q})$.*

3.2 ARV Structural Theorems

Having defined appropriate notations for SoS, we now move on to another crucial preliminary: the ARV Structural Theorem. It will be useful to state the theorem in terms of both metrics and pseudo-expectation. Let us start by definitions of several notations for metrics.

▶ **Definition 8** (Metric-Related Notations)**.** A *metric* d on X is a function $d : X \times X \to \mathbb{R}_{\geqslant 0}$ that satisfies (1) $d(x, x) = 0$, (2) symmetry $d(x, y) = d(y, x)$ and (3) triangle inequality $d(x, z) \leqslant d(x, y) + d(y, z)$, for all $x, y, z \in X$. We use the following notations in this work:

- For $x \in X$ and $S, T \subseteq X$, $d(x, S) := \min_{y \in S} d(x, y)$ and $d(S, T) := \min_{y \in S} d(y, T)$.
- We say that S, T are Δ-*separated* iff $d(S, T) \geqslant \Delta$.
- The *diameter* of a metric space (X, d) denoted by $\mathrm{diam}(X, d)$ is $\max_{x,y \in X} d(x, y)$.
- We say that (X, d) is α-spread if $\sum_{x,y \in X} d(x, y) \geqslant \alpha |X|^2$.
- An *(open) ball of radius r around x* denoted by $\mathcal{B}_d(x, r)$ is defined as $\{y \in X \mid d(x, y) < r\}$.
- A metric space (X, d) is said to be *(r, m)-hollow* if $|\mathcal{B}_d(x, r)| \leqslant m$ for all $x \in X$.

▶ **Definition 9** (Negative Type Metric)**.** A metric space (X, d) is said to be of *negative type* if $\sqrt{d}$ is Euclidean, i.e., for some $f : X \to \mathbb{R}^q$, $\|f(x) - f(y)\|_2^2 = d(x, y)$ for all $x, y \in X$.

The ARV Theorem can now be stated as follows:

▶ **Theorem 10** (ARV Structural Theorem - Metric Formulation [6, 42])**.** *Let $\alpha, r > 0$ be any positive real number and $m \in \mathbb{N}$ be any positive integer. For any negative type metric space (X, d) with $\mathrm{diam}(d) \leqslant 1$ that is α-spread and (r, m)-hollow, there exist $\Omega_{\alpha,\beta}(1/\sqrt{\log m})$-separated subsets $T, T' \subseteq X$ each of size $\Omega(|X|)$. Moreover, these sets can be found in randomized polynomial time.*

We remark that the quantitative bound $\Delta = \Omega_{\alpha,\beta}(1/\sqrt{\log m})$ follows from Lee's version of the theorem [42] whereas the original version only have $\Delta = \Omega_{\alpha,\beta}(1/(\log m)^{2/3})$.

As we are using the ARV Theorem in conjunction with the SoS conditioning framework, it is useful to also state the theorem in SoS notations. To do so, let us first state the following

fact, which can be easily seen via the fact that the moment matrix (with (i,j)-entry equal to $\tilde{\mathbb{E}}[X_iX_j]$) is positive semidefinite and thus is a Gram matrix for some set of vectors:

▶ **Proposition 11.** *Let $\tilde{\mathbb{E}} : \mathbb{R}_2[X_1,\dots,X_n] \to \mathbb{R}$ be any degree-2 p.e. that satisfies $\tilde{\mathbb{E}}[(X_i - X_j)^2] \leqslant \tilde{\mathbb{E}}[(X_i - X_k)^2] + \tilde{\mathbb{E}}[(X_k - X_j)^2]$ for all $i,j,k \in [n]$. Define $d_{\tilde{\mathbb{E}}} : [n] \times [n] \to \mathbb{R}_{\geqslant 0}$ by $d_{\tilde{\mathbb{E}}}(i,j) = \tilde{\mathbb{E}}[(X_i - X_j)^2]$. Then, $([n], d_{\tilde{\mathbb{E}}})$ is a negative type metric space.*

When it is clear which pseudo-expectation we are referring to, we may drop the subscript from $d_{\tilde{\mathbb{E}}}$ and simply write d. Further, we use all metric terminologies with $\tilde{\mathbb{E}}$ in the natural manner; for instance, we say that $S, T \subseteq [n]$ are Δ-separated if $d_{\tilde{\mathbb{E}}}(S,T) \geqslant \Delta$.

Theorem 10 can now be restated in pseudo-expectation notations as follows.

▶ **Theorem 12** (ARV Structural Theorem - SoS Formulation [6, 42])**.** *For any $\alpha, \beta > 0$ and $m \in \mathbb{N}$, let $\tilde{\mathbb{E}} : \mathbb{R}_2[X_1,\dots,X_n] \to \mathbb{R}$ be any degree-2 p.e. such that the following hold:*

- *(Boolean) For every $i \in [n]$, $\tilde{\mathbb{E}}[X_i^2] = 1$.*
- *(Triangle Inequality) For every $i,j,k \in [n]$, $\tilde{\mathbb{E}}[(X_i - X_j)^2] \leqslant \tilde{\mathbb{E}}[(X_i - X_k)^2] + \tilde{\mathbb{E}}[(X_k - X_j)^2]$.*
- *(Balance) $\sum_{i,j \in [n]} \tilde{\mathbb{E}}[(X_i - X_j)^2] \geqslant \alpha n^2$.*
- *(Hollowness) For all $i \in [n]$, $|\{j \in [n] \mid \tilde{\mathbb{E}}[X_iX_j] > 1 - \beta\}| \leqslant m$.*

Then, there exists a randomized polynomial time algorithm that, with probability 2/3, produces $\Omega_{\alpha,\beta}(1/\sqrt{\log m})$-separated subsets $T, T' \subseteq [n]$ each of size $\Omega_{\alpha,\beta}(n)$.

Notice that, for boolean $\tilde{\mathbb{E}}$, $\tilde{\mathbb{E}}[X_iX_j] = 1 - \tilde{\mathbb{E}}[(X_i - X_j)^2]/2 = 1 - d_{\tilde{\mathbb{E}}}(i,j)/2$. This means that $\{j \in [n] \mid \tilde{\mathbb{E}}[X_iX_j] > 1 - \beta\}$ is simply $\mathcal{B}_{d_{\tilde{\mathbb{E}}}}(i, 2\beta)$. Another point to notice is that the metric $d_{\tilde{\mathbb{E}}}$ can have $\operatorname{diam}(d_{\tilde{\mathbb{E}}})$ as large as 4, instead of 1 required in Theorem 10, but this poses no issue since we can scale all distances down by a factor of 4.

4 Conditioning Yields Easy-To-Round Solution

Our main tool is the following lemma on structure of conditioned solution:

▶ **Lemma 13.** *Let τ, γ be any positive real numbers such that $\tau^2 < \gamma < 1$. Given a boolean degree-d pseudo-expectation $\tilde{\mathbb{E}} : \mathbb{R}_d[X_1,\dots,X_n] \to \mathbb{R}$ for a system $(\mathcal{P}, \mathcal{Q})$ and an integer $\ell < d$, we can, in time $O(n/d)^{O(d)}$, find a boolean degree-$(d-\ell)$ pseudo-expectation $\tilde{\mathbb{E}}' : \mathbb{R}_{d-\ell}[X_1,\dots,X_n] \to \mathbb{R}$ for the system $(\mathcal{P}, \mathcal{Q})$ such that the following condition holds:*

- *Let $V_{(-\tau,\tau)} := \{i \in [n] \mid \tilde{\mathbb{E}}'[X_i] \in (-\tau,\tau)\}$ denote the set of indices of variables whose pseudo-expectation lies in $(-\tau,\tau)$ and, for each $i \in [n]$, let $C_\gamma(i) := \{j \in [n] \mid \tilde{\mathbb{E}}'[X_iX_j] \in [-\gamma,\gamma]\}$ denote the set of all indices j's such that $\tilde{\mathbb{E}}'[X_iX_j]$ lies in $[-\gamma,\gamma]$. Then, for all $i \in V_{(-\tau,\tau)}$, we have*

$$|V_{(-\tau,\tau)} \setminus C_\gamma(i)| \leqslant \frac{n}{\ell(\gamma - \tau^2)^2}.$$

In other words, the lemma says that, when d is sufficiently large, we can condition so that we arrive at a pseudo-expectation with the hollowness condition if we restrict ourselves to $V_{(-\tau,\tau)}$. Note here that, outside of $V_{(-\tau,\tau)}$, this hollowness condition does not necessarily hold. For instance, it could be that after conditioning all variables be come integral (i.e. $\tilde{\mathbb{E}}[X_i] \in \{\pm 1\}$). However, this is the second "easy-to-round" case for ARV theorem, so this does not pose a problem for us.

The proof of Lemma 13 is based on a potential function argument. In particular, the potential function we use is $\Phi(\tilde{\mathbb{E}}) = \sum_{i \in [n]} \tilde{\mathbb{E}}[X_i]^2$. The main idea is that, as long as there is a "bad" $i \in [n]$ that violates the condition in the lemma, we will be able to finding a conditioning that significantly increases Φ. However, Φ is always at most n, meaning that this cannot

happens too many times and, thus, we must at some point arrive at a pseudo-expectation with no bad i. Due to space constraint, we defer the full proof to the appendix.

5 The Algorithms

All of our algorithms follow the same three-step blueprint, as summarized below.

Step I: Solving for Degree-$n/2^{\Omega(r^2)}$ Pseudo-Expectation. We first consider the system of constraints corresponding to the best known existing polynomial time algorithm for each problem, and we solve for degree-$n/2^{\Omega(r^2)}$ pseudo-expectation for such a system.

Step II: Conditioning to Get "Hollow" Solution. Then, we apply Lemma 13 to arrive at a degree-2 pseudo-expectation that satisfies the system and that additionally is hollow, i.e., $|V_{(-\tau,\tau)} \setminus C_\gamma(i)| \leqslant 2^{r^2}$ for all $i \in V_{(-\tau,\tau)}$ for appropriate values of τ, γ. Recall here that $V_{(-\tau,\tau)}$ and $C_\gamma(i)$ are defined in Lemma 13.

Step III: Following the Existing Algorithm. Finally, we follow the existing polynomial time approximation algorithms [6, 32, 1] to arrive at an approximate solution for the problem of interest. The improvement in the approximation ratio comes from the fact that our pseudo-expectation is now in the "easy-to-round" regime, i.e., the ARV Theorem gives separation of $\Omega(1/r)$ for this regime instead of $\Omega(1/\sqrt{\log n})$ for the general regime.

Let us now demonstrate our framework with the Balanced Separator problem.

▶ **Theorem 14.** *For any $r > 1$ (possibly depending on n), there exists an $\exp(n/2^{\Omega(r^2)})$ poly(n)-time $O(r)$-approximation for Balanced Separator on n-vertex graphs.*

Proof. On input graph $G = (V = [n], E)$, the algorithm works as follows.

Step I: Solving for Degree-$n/2^{\Omega(r^2)}$ Pseudo-Expectation. For every real number $OBJ \in \mathbb{R}$, let $(\mathcal{P}^{BS}_{G,OBJ}, \mathcal{Q}^{BS}_{G,OBJ})$ be the following system of equations:

1. (Boolean) For all $i \in [n]$, $X_i^2 - 1 = 0$.
2. (Balance) $\sum_{i,j\in[n]}(X_i - X_j)^2 - 16n^2/9 \geqslant 0, n/3 - \sum_{i\in[n]} X_i \geqslant 0$ and $\sum_{i\in[n]} X_i + n/3 \geqslant 0$.
3. (Triangle Inequalities) For all $i, j, k \in [n]$, $(X_i - X_k)^2 + (X_k - X_j)^2 - (X_i - X_j)^2 \geqslant 0, (1 - X_i)^2 + (1 - X_j)^2 - (X_i - X_j)^2 \geqslant 0$, and $(1 + X_i)^2 + (1 - X_j)^2 - (X_i - X_j)^2 \geqslant 0$.
4. (Objective Bound) $4 \cdot OBJ - \sum_{(i,j)\in E}(X_i - X_j)^2 \geqslant 0$.

Let $D := \lceil 1000n/2^{r^2} \rceil + 2$. The algorithm first uses binary search to find the largest OBJ such that there exists a degree-D pseudo-expectation for $(\mathcal{P}^{BS}_{G,OBJ}, \mathcal{Q}^{BS}_{G,OBJ})$. Let this value of OBJ be OBJ^*, and let $\tilde{\mathbb{E}}$ be a degree-D pseudo-expectation satisfying $(\mathcal{P}^{BS}_{G,OBJ^*}, \mathcal{Q}^{BS}_{G,OBJ^*})$.

Again, observe that this step takes $\exp\left(n/2^{\Omega(r^2)}\right)\text{poly}(n)$ time and $OBJ^* \leqslant OPT$ where OPT is the number of edges cut in the balanced separator of G.

Step II: Conditioning to Get "Hollow" Solution. Use Lemma 13 to find a degree-2 p.e. $\tilde{\mathbb{E}}'$ for $(\mathcal{P}^{BS}_{G,OBJ^*}, \mathcal{Q}^{BS}_{G,OBJ^*})$ such that for all $i \in V_{(-0.9,0.9)}$, $|V_{(-0.9,0.9)} \setminus C_{0.9}(i)| < 2^{r^2}$.

Step III: Following ARV Algorithm. The last step follows the ARV algorithm [6], which first uses the ARV lemma to obtain two large well separated set. While in the traditional setting, the structural theorem can be applied immediately; we have to be more careful and treat the two "easy-to-round" cases differently. This is formalized below.

▶ **Claim 15.** *There exist disjoint subsets $T, T' \subseteq [n]$ that are $\Omega(1/r)$-separated. Moreover, these subsets can be found (with probability 2/3) in polynomial time.*

Proof. For every $a, b \in \mathbb{R}$, let $V_{\geqslant a} = \{i \in [n] \mid \tilde{\mathbb{E}}'[X_i] \geqslant a\}$, $V_{\leqslant b} = \{i \in [n] \mid \tilde{\mathbb{E}}'[X_i] \leqslant b\}$ and $V_{(a,b)} = \{i \in [n] \mid a < \tilde{\mathbb{E}}'[X_i]| < b\}$. Let $\tau = 0.9$. We consider the following two cases:

1. $|V_{\geqslant \tau}| \geqslant 0.1n$ or $|V_{\leqslant -\tau}| \geqslant 0.1n$. Suppose without loss of generality that it is the former. We claim that $|V_{\leqslant 0.8}| \geqslant 0.2n$. To see that this is the case, observe that

$$n/3 = \sum_{i \in V \setminus V_{\leqslant 0.8}} \tilde{\mathbb{E}}'[X_i] + \sum_{i \in V_{\leqslant 0.8}} \tilde{\mathbb{E}}'[X_i] \quad \geqslant 0.8(n - |V_{\leqslant 0.8}|) - |V_{\leqslant 0.8}| = 0.8n - 1.8|V_{\leqslant 0.8}|$$

which implies that $|V_{\leqslant 0.8}| > 0.2n$. Let $T = V_{\geqslant \tau}$ and $T' = V_{\leqslant 0.8}$. As we have shown, $|T|, |T'| \geqslant \Omega(n)$. Moreover, for every $i \in T$ and $j \in T'$, triangle inequality implies that

$$\tilde{\mathbb{E}}'[X_i X_j] \leqslant \tilde{\mathbb{E}}'[X_i X_j] + \tilde{\mathbb{E}}'[(1 - X_i)(1 + X_j)] = 1 - \tilde{\mathbb{E}}'[X_i] + \tilde{\mathbb{E}}'[X_j] < 1 - 0.9 + 0.8 = 0.9.$$

That is, $\tilde{\mathbb{E}}'[(X_i - X_j)^2] = 2 - 2\tilde{\mathbb{E}}'[X_i X_j] > 0.2$, completing the proof for the first case.

2. $|V_{\geqslant \tau}| < 0.1n$ and $|V_{\leqslant -\tau}| < 0.1n$. This implies that $|V_{(-\tau,\tau)}| \geqslant 0.8n$. Moreover,

$$\sum_{i,j \in V_{(-\tau,\tau)}} \tilde{\mathbb{E}}[(X_i - X_j)^2] \geqslant 16n^2/9 - 8n|V_{\geqslant \tau}| - 8n|V_{\leqslant \tau}| > 0.1n^2.$$

Hence, applying the ARV Theorem (Theorem 12) to $V_{(-\tau,\tau)}$ yields the desired T, T'. Thus, in both cases, we can find the desired T, T' in randomized polynomial time. ◀

Once we have found T, T', we use the following rounding scheme from [43, 6]: pick θ uniformly at random from $[0, d(T, T'))$. Then, output $(S, V \setminus S)$ where $S = \{i \in [n] \mid d(i, T) < \theta\}$.

Observe that $T \subseteq S$ and $T' \subseteq (V \setminus S)$, which means that $|S|, |V \setminus S| > \Omega(n)$; in other words, the output is a valid solution for the problem. Moreover, for every $(i, j) \in E$, it is easy to see that the probability that i and j end up in different sets is at most $|d(i, T) - d(j, T)|/d(T, T') \leqslant d(i, j)/d(T, T') \leqslant O(r) \cdot d(i, j)$. As a result, the expected number of edges cut is $O(r) \cdot \sum_{(i,j) \in E} d(i, j) = O(r) \cdot OBJ^*$, which completes our proof. ◀

6 Conclusion and Open Questions

In this work, we use the conditioning framework in the SoS Hierarchy together with the ARV Theorem to design "fast" exponential time approximation algorithms for Vertex Cover, Uniform Sparsest Cut and Balanced Separator that achieve significant speed-up over the trivial "limited brute force" algorithms. While we view this as a step towards understanding the time vs approximation ratio trade-off for these problems, many questions remain open.

First, as discussed in the introduction, current lower bounds do not rule out subexponential time approximation algorithms in the regime of our study. For instance, an 1.9-approximation algorithm for Vertex Cover could still possibly be achieved in say $2^{O(\sqrt{n})}$ time. Similarly for Uniform Sparsest Cut and Balanced Separator, $O(1)$-approximation for them could still possibly be achieved in subexponential time. The main open question is to either confirm that such algorithms exist, or rule them out under certain believable complexity hypotheses.

Another, perhaps more plausible, direction is to try to extend our technique to other problems for which the known polynomial time approximation algorithms employ the ARV Theorem. The most obvious candidates are Minimum UnCut, Minimum 2CNF Deletion, Directed Balanced Separator and Directed Sparsest Cut, which were shown in [1] to admit $O(\sqrt{\log n})$-approximation in polynomial time. While not yet rigorously verified, we believe

that our technique also yields approximation algorithms with similar running time as Uniform Sparsest Cut to these problems as well. In this direction, there are also other potentially more challenging candidates, such as Balanced Vertex Separator, (Non-uniform) Sparsest Cut, and Minimum Linear Arrangement. While the first problem admits $O(\sqrt{\log n})$-approximation in polynomial time [23], additional ingredients beyond the ARV Theorem are required in the algorithm. On the other hand, the latter two problems only admit $O(\sqrt{\log n} \log \log n)$-approximation [4, 16, 24]. It seems challenging to remove this $\log \log n$ factor and achieve a constant factor approximation, even in our "fast" exponential time regime.

References

1. Amit Agarwal, Moses Charikar, Konstantin Makarychev, and Yury Makarychev. $O(\sqrt{\log n})$ approximation algorithms for min UnCut, min 2CNF deletion, and directed cut problems. In *STOC*, pages 573–581, 2005. doi:10.1145/1060590.1060675.
2. Sanjeev Arora, Boaz Barak, and David Steurer. Subexponential algorithms for unique games and related problems. *J. ACM*, 62(5):42:1–42:25, 2015. doi:10.1145/2775105.
3. Sanjeev Arora, Russell Impagliazzo, William Matthews, and David Steurer. Improved algorithms for unique games via divide and conquer. *ECCC*, 17:41, 2010.
4. Sanjeev Arora, James R. Lee, and Assaf Naor. Euclidean distortion and the sparsest cut. In *STOC*, pages 553–562, 2005. doi:10.1145/1060590.1060673.
5. Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *J. ACM*, 45(3):501–555, 1998. doi:10.1145/278298.278306.
6. Sanjeev Arora, Satish Rao, and Umesh V. Vazirani. Expander flows, geometric embeddings and graph partitioning. *J. ACM*, 56(2):5:1–5:37, 2009. doi:10.1145/1502793.1502794.
7. Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of NP. *J. ACM*, 45(1):70–122, 1998. doi:10.1145/273865.273901.
8. Per Austrin, Siavosh Benabbas, and Konstantinos Georgiou. Better balance by being biased: A 0.8776-approximation for max bisection. In *SODA*, pages 277–294, 2013. doi:10.1137/1.9781611973105.21.
9. Nikhil Bansal, Parinya Chalermsook, Bundit Laekhanukit, Danupon Nanongkai, and Jesper Nederlof. New tools and connections for exponential-time approximation. *CoRR*, abs/1708.03515, 2017. arXiv:1708.03515.
10. Nikhil Bansal and Subhash Khot. Optimal long code test with one free bit. In *FOCS*, pages 453–462, 2009. doi:10.1109/FOCS.2009.23.
11. R. Bar-Yehuda and S. Even. A local-ratio theorem for approximating the weighted vertex cover problem. In G. Ausiello and M. Lucertini, editors, *Analysis and Design of Algorithms for Combinatorial Problems*, volume 109 of *North-Holland Mathematics Studies*, pages 27–45. North-Holland, 1985. doi:10.1016/S0304-0208(08)73101-3.
12. Boaz Barak, Fernando G. S. L. Brandão, Aram Wettroth Harrow, Jonathan A. Kelner, David Steurer, and Yuan Zhou. Hypercontractivity, sum-of-squares proofs, and their applications. In *STOC*, pages 307–326, 2012. doi:10.1145/2213977.2214006.
13. Boaz Barak, Prasad Raghavendra, and David Steurer. Rounding semidefinite programming hierarchies via global correlation. In *FOCS*, pages 472–481, 2011. doi:10.1109/FOCS.2011.95.
14. Boaz Barak and David Steurer. Sum-of-squares proofs and the quest toward optimal algorithms. *ECCC*, 21:59, 2014. URL: http://eccc.hpi-web.de/report/2014/059.
15. Nicolas Bourgeois, Bruno Escoffier, and Vangelis Th. Paschos. Approximation of max independent set, min vertex cover and related problems by moderately exponential algorithms.

Discrete Applied Mathematics, 159(17):1954–1970, 2011. doi:10.1016/j.dam.2011.07.009.

16 Moses Charikar, Mohammad Taghi Hajiaghayi, Howard J. Karloff, and Satish Rao. ℓ_2^2 spreading metrics for vertex ordering problems. *Algorithmica*, 56(4):577–604, 2010. doi:10.1007/s00453-008-9191-1.

17 Moses Charikar, Konstantin Makarychev, and Yury Makarychev. Local global tradeoffs in metric embeddings. *SIAM J. Comput.*, 39(6):2487–2512, 2010. doi:10.1137/070712080.

18 Shuchi Chawla, Robert Krauthgamer, Ravi Kumar, Yuval Rabani, and D. Sivakumar. On the hardness of approximating multicut and sparsest-cut. *Computational Complexity*, 15(2):94–114, 2006. doi:10.1007/s00037-006-0210-9.

19 Irit Dinur, Subhash Khot, Guy Kindler, Dor Minzer, and Muli Safra. Towards a proof of the 2-to-1 games conjecture? *ECCC*, 23:198, 2016. URL: http://eccc.hpi-web.de/report/2016/198.

20 Irit Dinur, Subhash Khot, Guy Kindler, Dor Minzer, and Muli Safra. On non-optimally expanding sets in grassmann graphs. *ECCC*, 24:94, 2017. URL: https://eccc.weizmann.ac.il/report/2017/094.

21 Irit Dinur and Shmuel Safra. On the hardness of approximating minimum vertex cover. *Annals of Mathematics*, 162(1):439–485, 2005.

22 Uriel Feige. Relations between average case complexity and approximation complexity. In *STOC*, pages 534–543, 2002. doi:10.1145/509907.509985.

23 Uriel Feige, MohammadTaghi Hajiaghayi, and James R. Lee. Improved approximation algorithms for minimum weight vertex separators. *SIAM J. Comput.*, 38(2):629–657, 2008. doi:10.1137/05064299X.

24 Uriel Feige and James R. Lee. An improved approximation ratio for the minimum linear arrangement problem. *Inf. Process. Lett.*, 101(1):26–29, 2007. doi:10.1016/j.ipl.2006.07.009.

25 Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

26 Dima Grigoriev. Complexity of positivstellensatz proofs for the knapsack. *Computational Complexity*, 10(2):139–154, 2001. doi:10.1007/s00037-001-8192-0.

27 Venkatesan Guruswami and Ali Kemal Sinop. Lasserre hierarchy, higher eigenvalues, and approximation schemes for graph partitioning and quadratic integer programming with PSD objectives. In *FOCS*, pages 482–491, 2011. doi:10.1109/FOCS.2011.36.

28 Eran Halperin. Improved approximation algorithms for the vertex cover problem in graphs and hypergraphs. *SIAM J. Comput.*, 31(5):1608–1623, 2002. doi:10.1137/S0097539700381097.

29 Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001. doi:10.1145/502090.502098.

30 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.

31 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.

32 George Karakostas. A better approximation ratio for the vertex cover problem. *ACM Trans. Algorithms*, 5(4):41:1–41:8, 2009. doi:10.1145/1597036.1597045.

33 Subhash Khot. On the power of unique 2-prover 1-round games. In *CCC*, page 25, 2002. doi:10.1109/CCC.2002.1004334.

34 Subhash Khot. Ruling out PTAS for graph min-bisection, dense k-subgraph, and bipartite clique. *SIAM J. Comput.*, 36(4):1025–1071, 2006. doi:10.1137/S0097539705447037.

35 Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell. Optimal inapproximability results for MAX-CUT and other 2-variable csps? *SIAM J. Comput.*, 37(1):319–357, 2007. doi:10.1137/S0097539705447372.

36 Subhash Khot, Dor Minzer, and Muli Safra. On independent sets, 2-to-2 games, and grassmann graphs. In *STOC*, pages 576–589, 2017. doi:10.1145/3055399.3055432.

37 Subhash Khot, Dor Minzer, and Muli Safra. Pseudorandom sets in grassmann graph have near-perfect expansion. *ECCC*, 25:6, 2018. URL: https://eccc.weizmann.ac.il/report/2018/006.

38 Subhash Khot and Oded Regev. Vertex cover might be hard to approximate to within 2-epsilon. *J. Comput. Syst. Sci.*, 74(3):335–349, 2008. doi:10.1016/j.jcss.2007.06.019.

39 Subhash Khot and Nisheeth K. Vishnoi. The unique games conjecture, integrality gap for cut problems and embeddability of negative-type metrics into ℓ_1. *J. ACM*, 62(1):8:1–8:39, 2015. doi:10.1145/2629614.

40 Pravesh K. Kothari, Ryuhei Mori, Ryan O'Donnell, and David Witmer. Sum of squares lower bounds for refuting any CSP. In *STOC*, pages 132–145, 2017.

41 Jean B. Lasserre. An explicit equivalent positive semidefinite program for nonlinear 0-1 programs. *SIAM Journal on Optimization*, 12(3):756–769, 2002. doi:10.1137/S1052623400380079.

42 James R. Lee. On distance scales, embeddings, and efficient relaxations of the cut cone. In *SODA*, pages 92–101, 2005. URL: http://dl.acm.org/citation.cfm?id=1070432.1070446.

43 Frank Thomson Leighton and Satish Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *J. ACM*, 46(6):787–832, 1999. doi:10.1145/331524.331526.

44 Pasin Manurangsi and Prasad Raghavendra. A birthday repetition theorem and complexity of approximating dense CSPs. *CoRR*, abs/1607.02986, 2016. URL: http://arxiv.org/abs/1607.02986, arXiv:1607.02986.

45 Burkhard Monien and Ewald Speckenmeyer. Ramsey numbers and an approximation algorithm for the vertex cover problem. *Acta Inf.*, 22(1):115–123, 1985. doi:10.1007/BF00290149.

46 Dana Moshkovitz and Ran Raz. Two-query PCP with subconstant error. *J. ACM*, 57(5):29:1–29:29, 2010. doi:10.1145/1754399.1754402.

47 Yurii Nesterov. Squared functional systems and optimization problems. In *High performance optimization*, pages 405–440. Springer, 2000.

48 Ryan O'Donnell and Yuan Zhou. Approximability and proof complexity. In *SODA*, pages 1537–1556, 2013. doi:10.1137/1.9781611973105.111.

49 Pablo A. Parrilo. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. PhD thesis, California Institute of Technology, 2000.

50 Prasad Raghavendra and David Steurer. Graph expansion and the unique games conjecture. In *STOC*, pages 755–764, 2010. doi:10.1145/1806689.1806788.

51 Prasad Raghavendra, David Steurer, and Madhur Tulsiani. Reductions between expansion problems. In *CCC*, pages 64–73, 2012. doi:10.1109/CCC.2012.43.

52 Prasad Raghavendra and Ning Tan. Approximating CSPs with global cardinality constraints using SDP hierarchies. In *SODA*, pages 373–387, 2012. URL: http://portal.acm.org/citation.cfm?id=2095149&CFID=63838676&CFTOKEN=79617016.

53 Grant Schoenebeck. Linear level lasserre lower bounds for certain k-csps. In *FOCS*, pages 593–602, 2008. doi:10.1109/FOCS.2008.74.

54 Yuichi Yoshida and Yuan Zhou. Approximation schemes via sherali-adams hierarchy for dense constraint satisfaction problems and assignment problems. In *ITCS*, pages 423–438, 2014. doi:10.1145/2554797.2554836.

A Proof of the Conditioning Lemma

In this section, we prove Lemma 13. To facilitate our proof, let us prove a simple identity regarding the potential change for a single variable after conditioning:

▶ **Proposition 16.** *Let $\tilde{\mathbb{E}} : \mathbb{R}_d[X_1, \ldots, X_n] \to \mathbb{R}$ be any degree-d pseudo-expectation for some $d > 2$ and let $i \in [n]$ be such that $\tilde{\mathbb{E}}[X_i] \neq -1, 1$. Then, for any $j \in [n]$, we have*

$$\left(\frac{1-\tilde{\mathbb{E}}[X_i]}{2}\right)\left(\tilde{\mathbb{E}}|_{X_i=-1}[X_j]\right)^2 + \left(\frac{1+\tilde{\mathbb{E}}[X_i]}{2}\right)\left(\tilde{\mathbb{E}}|_{X_i=1}[X_j]\right)^2 - \tilde{\mathbb{E}}[X_j]^2 = \frac{(\tilde{\mathbb{E}}[X_iX_j] - \tilde{\mathbb{E}}[X_i]\tilde{\mathbb{E}}[X_j])^2}{1-\tilde{\mathbb{E}}[X_i]^2}.$$

Proof. For succinctness, let $a = (1-\tilde{\mathbb{E}}[X_i])/2, b = \tilde{\mathbb{E}}|_{X_i=1}[X_j]$ and $c = \tilde{\mathbb{E}}|_{X_i=-1}[X_j]$. Observe that, from definition of conditioning, we have

$$ab + (1-a)c = \tilde{\mathbb{E}}[(1-X_i)X_j/2] + \tilde{\mathbb{E}}[(1+X_i)X_j/2] = \tilde{\mathbb{E}}[X_j].$$

Hence, the left hand side term of the equation in the proposition statement can be rewritten as

$$ab^2 + (1-a)c^2 - (ab + (1-a)c)^2 = a(1-a)(b-c)^2. \tag{6}$$

Let $\mu_i = \tilde{\mathbb{E}}[X_i]$. Now, observe that $b - c$ is simply

$$\begin{aligned}
\tilde{\mathbb{E}}|_{X_i=-1}[X_j] - \tilde{\mathbb{E}}|_{X_i=1}[X_j] &= \frac{\tilde{\mathbb{E}}[(1-X_i)X_j]}{1-\mu_i} - \frac{\tilde{\mathbb{E}}[(1+X_i)X_j]}{1+\mu_i} \\
&= \frac{\tilde{\mathbb{E}}[(1+\mu_i)(1-X_i)X_j - (1-\mu_i)(1+X_i)X_j]}{1-\mu_i^2} \\
&= \frac{\tilde{\mathbb{E}}[2(\mu_i - X_i)X_j]}{1-\mu_i^2} \\
&= \frac{2(\mu_i\tilde{\mathbb{E}}[X_j] - \tilde{\mathbb{E}}[X_iX_j])}{1-\mu_i^2}.
\end{aligned}$$

Plugging the above equality back into (6) yields the desired identity. ◀

With the above lemma ready, we now proceed to the proof of Lemma 13. Before we do so, let us also note that our choice of potential function $\tilde{\mathbb{E}}[X_i]^2$ is not of particular importance; indeed, there are many other potential functions that work, such as the entropy of X_i.

Proof of Lemma 13. We describe an algorithm below that finds $\tilde{\mathbb{E}}'$ by iteratively conditioning the pseudo-distribution on the variable X_i that violates the condition.

1. Let $\tilde{\mathbb{E}}_0 = \tilde{\mathbb{E}}$
2. For $t = 1, \ldots, \ell$, execute the following steps.
   a. Let $V^{t-1}_{(-\tau,\tau)} := \{i \in [n] \mid \tilde{\mathbb{E}}_{t-1}[X_i] \in (-\tau, \tau)\}$.
   Moreover, for each $i \in [n]$, let $C^{t-1}_{\gamma}(i) := \{j \in [n] \mid \tilde{\mathbb{E}}_{t-1}[X_iX_j] \in [-\gamma, \gamma]\}$.
   b. If $|V^{t-1}_{(-\tau,\tau)} \setminus C^{t-1}_{\gamma}(i)| \leqslant \frac{n}{\ell(\gamma-\tau^2)^2}$ for all $i \in V^{t-1}_{(-\tau,\tau)}$, then output $\tilde{\mathbb{E}}_{t-1}$ and terminate.
   c. Otherwise, pick $i \in V^{t-1}_{(-\tau,\tau)}$ such that $|V^{t-1}_{(-\tau,\tau)} \setminus C^{t-1}_{\gamma}(i)| > \frac{n}{\ell(\gamma-\tau^2)^2}$. Compute $\Phi(\tilde{\mathbb{E}}_t|_{X_i=1})$ and $\Phi(\tilde{\mathbb{E}}_t|_{X_i=-1})$ and let $\tilde{\mathbb{E}}_t$ be equal to the one with larger potential.
3. If the algorithm has not terminated, output NULL.

Notice that, if the algorithm terminates in Step 2b, then the output pseudo-distribution obviously satisfies the condition in Lemma 13. Hence, we only need to show that the algorithm always terminates in Step 2b (and never reaches Step 3). Recall that we let $\Phi(\tilde{\mathbb{E}})$ denote $\sum_{i\in[n]} \tilde{\mathbb{E}}[X_i]^2$. To prove this, we will analyze the change in $\Phi(\tilde{\mathbb{E}}_t)$ over time. In particular, we can show the following:

▶ **Claim 17.** *For every $t \in [\ell]$, $\Phi(\tilde{\mathbb{E}}_t) - \Phi(\tilde{\mathbb{E}}_{t-1}) > n/\ell$.*

Proof. First, notice that it suffices to prove the following because $\tilde{\mathbb{E}}_{t-1}[(1-X_i)/2]+\tilde{\mathbb{E}}_{t-1}[(1+X_i)/2] = 1$ and, from our choice of $\tilde{\mathbb{E}}_t$, we have $\Phi(\tilde{\mathbb{E}}_t) = \max\{\Phi(\tilde{\mathbb{E}}_{t-1}|_{X_i=1}), \Phi(\tilde{\mathbb{E}}_{t-1}|_{X_i=-1})\}$.

$$\left(\tilde{\mathbb{E}}\left[\frac{1-X_i}{2}\right]\cdot\Phi(\tilde{\mathbb{E}}_{t-1}|_{X_i=-1}) + \tilde{\mathbb{E}}\left[\frac{1+X_i}{2}\right]\cdot\Phi(\tilde{\mathbb{E}}_{t+1}|_{X_i=1})\right) - \Phi(\tilde{\mathbb{E}}_{t-1}) > n/\ell.$$

Recall that, from our definition of Φ, the left hand side above can simply be written as

$$\sum_{j\in[n]}\left(\tilde{\mathbb{E}}\left[\frac{1-X_i}{2}\right]\left(\tilde{\mathbb{E}}_{t-1}|_{X_i=-1}[X_j]\right)^2 + \tilde{\mathbb{E}}\left[\frac{1+X_i}{2}\right]\left(\tilde{\mathbb{E}}_{t-1}|_{X_i=1}[X_j]\right)^2 - \tilde{\mathbb{E}}_{t-1}[X_j]^2\right).$$

From Proposition 16, this is equal to

$$\begin{aligned}\sum_{j\in[n]}\frac{\left(\tilde{\mathbb{E}}_{t-1}[X_iX_j]-\tilde{\mathbb{E}}_{t-1}[X_i]\tilde{\mathbb{E}}_{t-1}[X_j]\right)^2}{1-\tilde{\mathbb{E}}_{t-1}[X_i]^2} &\geqslant \sum_{j\in V^{t-1}_{(-\tau,\tau)}\setminus C^{t-1}_\gamma(i)}\frac{\left(\tilde{\mathbb{E}}_{t-1}[X_iX_j]-\tilde{\mathbb{E}}_{t-1}[X_i]\tilde{\mathbb{E}}_{t-1}[X_j]\right)^2}{1-\tilde{\mathbb{E}}_{t-1}[X_i]^2}\\ &> \sum_{j\in V^{t-1}_{(-\tau,\tau)}\setminus C^{t-1}_\gamma(i)}\left(\gamma-\tau^2\right)^2\\ \left(\text{From } |V^{t-1}_{(-\tau,\tau)}\setminus C^{t-1}_\gamma(i)| \geqslant \frac{n}{\ell(\gamma-\tau^2)^2}\right) &> n/\ell,\end{aligned}$$

where the second inequality follows from $|\tilde{\mathbb{E}}[X_i]|, |\tilde{\mathbb{E}}[X_j]| < \tau$ and $|\tilde{\mathbb{E}}[X_iX_j]| > \gamma$ for all $j \in V^{t-1}_{(-\tau,\tau)} \setminus C^{t-1}_\gamma(i)$. ◀

It is now easy to see that Claim 17 implies that the algorithm never reaches Step 3. Otherwise, we would have $\Phi(\tilde{\mathbb{E}}_\ell) > n/\ell + \Phi(\tilde{\mathbb{E}}_{\ell-1}) > \cdots > n + \Phi(\tilde{\mathbb{E}}) > n$, a contradiction. ◀

Deterministic $O(1)$-Approximation Algorithms to 1-Center Clustering with Outliers

Shyam Narayanan
Harvard University, Cambridge, Massachusetts, USA
shyamnarayanan@college.harvard.edu

Abstract

The *1-center clustering with outliers* problem asks about identifying a prototypical robust statistic that approximates the location of a cluster of points. Given some constant $0 < \alpha < 1$ and n points such that αn of them are in some (unknown) ball of radius r, the goal is to compute a ball of radius $O(r)$ that also contains αn points. This problem can be formulated with the points in a normed vector space such as $\mathbb{R}^d$ or in a general metric space.

The problem has a simple randomized solution: a randomly selected point is a correct solution with constant probability, and its correctness can be verified in linear time. However, the *deterministic* complexity of this problem was not known. In this paper, for any L^p *vector space*, we show an $O(nd)$-time solution with a ball of radius $O(r)$ for a fixed $\alpha > \frac{1}{2}$, and for any *normed vector space*, we show an $O(nd)$-time solution with a ball of radius $O(r)$ when $\alpha > \frac{1}{2}$ as well as an $O(nd \log^{(k)}(n))$-time solution with a ball of radius $O(r)$ for all $\alpha > 0, k \in \mathbb{N}$, where $\log^{(k)}(n)$ represents the kth iterated logarithm, assuming distance computation and vector space operations take $O(d)$ time. For an *arbitrary metric space*, we show for any $C \in \mathbb{N}$ an $O(n^{1+1/C})$-time solution that finds a ball of radius $2Cr$, assuming distance computation between any pair of points takes $O(1)$-time, and show that for any α, C, an $O(n^{1+1/C})$-time solution that finds a ball of radius $((2C-3)(1-\alpha)-1)r$ cannot exist.

2012 ACM Subject Classification Theory of computation → Facility location and clustering; Theory of computation → Divide and conquer

Keywords and phrases Deterministic, Approximation Algorithm, Cluster, Statistic

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.21

Acknowledgements I want to thank Professors Piotr Indyk and Jelani Nelson, who proposed this problem in a course I took with them. I would also like to thank them both for providing a lot of feedback on my work and write-up. I would also like to thank James Tao for a helpful discussion.

1 Introduction

Data clustering that is tolerant to outliers is a well-studied task in machine learning and computational statistics. In this paper, we deal with one of the simplest examples of this class of problems: *1-center clustering with outliers*. Informally, given n points such that there exists an unknown ball of radius r containing most of the points, we wish to find a ball of radius $O(r)$ also containing a large fraction of the points. More formally, suppose $0 < \alpha < 1$ is some fixed constant. Given points $a_1, ..., a_n$ in space $\mathbb{R}^d$ (where points are given as coordinates) under an L^p norm for some $p \geq 1$, in some other normed vector space, or in an arbitrary metric space (where we just have access to distances), suppose we know there exists a ball of radius r containing at least αn points but do not know the location of the ball. Then, can we efficiently provide a C-approximation to finding the ball, i.e. find the center of a ball of radius Cr for some $C \geq 1$ containing at least αn points?

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018).
Editors: Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer; Article No. 21; pp. 21:1–21:19

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

The problem has a simple linear-time Las Vegas randomized algorithm: a randomly selected point is a correct solution with constant probability, and its correctness can be verified in linear time. In fact, an even faster randomized algorithm works by picking $O(1)$ points randomly, computing pairwise distances, and selecting a cluster if it exists. However, the *deterministic* complexity of this problem appears more intriguing, and to the best of our knowledge, no linear-time or even subquadratic-time (let alone simple) solution for this problem was known. A trivial quadratic-time algorithm exists by enumerating over all points and checking pairwise distances, so the goal of the paper is to obtain deterministic algorithms whose running time is faster than the above. This situation bears similarity to the closely related *1-median* problem, where given a set of points $a_1, ..., a_n$ we want to find a point p^* that (approximately) minimizes the sum of the distances between p^* and all a_i's. It is a folklore fact that a randomly selected point is a $2(1+\epsilon)$-approximate 1-median with probability at least $\frac{\epsilon}{1+\epsilon}$. However, in the deterministic case for an arbitrary metric space, no constant-factor approximation in linear time is possible [7, 5], and non-trivial tradeoffs between the approximation factor and the running time exist [6, 4]. The goal of this paper is to establish an analogous understanding of the deterministic complexity of 1-center clustering with outliers.

1.1 Main results

Our results are depicted in Table 1. They primarily fall into two main categories: results in normed vector spaces and results in arbitrary metric spaces. For $\mathbb{R}^d$ with the L^p norm, assuming we are given coordinates of points, our algorithm runs in $O(nd)$ time with an $O((\alpha - 0.5)^{-1/p})$-approximation, assuming $\alpha > \frac{1}{2}$. Such a runtime even for the Euclidean case was previously unknown. For arbitrary normed vector spaces, our algorithm runs in $O_\alpha(nd)$ time with an $O((\alpha - 0.5)^{-1})$-approximation whenever $\alpha > 0.5$, assuming that distance calculation, vector addition, and vector multiplication can be done in $O(d)$ time. For $0 < \alpha \leq 0.5$, we solve the problem for arbitrary normed vector spaces in $O_{\alpha,k}(nd \log^{(k)}(n)) = O_{\alpha,k}(nd \log\log ... \log n)$ time for any integer k.

For arbitrary metric spaces, assuming distance calculation takes $O(1)$ time, we give an $O_{\alpha,C}(n^{1+1/C})$-time algorithm with approximation constant $2C$. While this is much weaker than for normed vector spaces, it is not possible to do much better, as for any fixed α and C, there is no $O(n^{1+1/C})$-time algorithm with approximation constant $(2C-3)(1-\alpha)-1$ that works for an arbitrary metric space. In particular, there is no $O(n \text{ polylog } n)$-time solution to solve the general metric space problem, even for large α. See Appendix A for the proof.

As a note, subscripts of α, k, and C on our O factors mean that the constants may depend on α, k, and C but are at most some increasing function of the subset of α^{-1}, k, C in the subscript.

1.2 Motivation and Relation to Previous Work

1-center clustering with outliers is a very simple example of a robust statistic, i.e. its location is usually resistant to large changes to a small fraction of the data points. Robust statistics are reviewed in detail in [14]. When $\alpha > \frac{1}{2}$, addition of a large number of points does not change the statistic up to $O(r)$, as it only slightly decreases the value of α. Even if $\alpha < \frac{1}{2}$, the statistic is still robust as if we find some ball containing αn points that are disjoint from the intended ball, we can remove those points and now there is some ball with at least $\alpha' = \frac{\alpha}{1-\alpha}$ of the remaining points which we need to get close to, so inducting on $\lfloor \alpha^{-1} \rfloor$ shows that the statistic is robust.

Table 1 Our results.

Space	Assumptions	Runtime	Approximation	Comments
L^p normed	$\alpha > \frac{1}{2}$	$O(nd)$	$O\left((\alpha - 0.5)^{-1/p}\right)$	Implies Euclidean
Normed	$\alpha > \frac{1}{2}$	$O_\alpha(nd)$	$O\left((\alpha - 0.5)^{-1}\right)$	
Normed	$\alpha > 0$	$O_{\alpha,k}(nd \log^{(k)}(n))$	$O_{\alpha,k}(1)$	Implies for L^p space, k any positive integer
Metric	$\alpha > 0$	$O_{\alpha,C}(n^{1+1/C})$	$2C$	Can be done even if the radius is unknown, C any positive integer
Metric	$\alpha > 0$	$\omega(n^{1+1/C})$	$(2C-3)(1-\alpha)-1$	Reduction from metric 1-median lower bound, see Appendix A

Robust statistics have a lot of practical use in statistics and machine learning [9, 13]. Since machine learning often deals with large amounts of data, it is difficult to obtain a large amount of data with high accuracy in a short period of time. Therefore, if we can compute a robust statistic quickly, we can get more data in the same amount of time and have a good understanding of the approximate location of a good fraction of the data.

This question is valuable from the perspective of derandomization. One solution to the 1-center clustering problem is to randomly select a point and check if it is at most $2r$ away from $\alpha n - 1$ other points, and repeat the process if it fails. This algorithm is efficient and gets a ball of radius $2r$ with αn points after $O(\alpha^{-1}n)$ expected computations, but is a Las Vegas algorithm that can be slow with reasonable probability. A faster Monte Carlo algorithm involves choosing an $O(1)$-size subset of the points and running the brute force quadratic algorithm, though similarly this algorithm may fail with reasonable probability. Therefore, this problem relates to the question of the extent to which randomness is required to solve certain computational problems.

The Euclidean problem is useful in the amplification of an Approximate Matrix Multiplication (AMM) algorithm described in [11]. To compute A^TB up to low Frobenius norm error with probability $2/3$ in low time and space, the algorithm approximates A^TB as $C = (SA)^T(SB)$, where S is a certain randomized sketch matrix. Then, if this process is repeated $O(\log \delta^{-1})$ times to get $C_1, ..., C_{O(\log \delta^{-1})}$, with probability $1 - \delta$, at least $3/5$ of the C_i's satisfy $||C_i - A^TB||_F \leq \epsilon ||A||_F ||B||_F$. We are able to approximate $||A||_F ||B||_F$ with high probability using L_2 approximation algorithms from [1]. If we think of C_i and A^TB as vectors, at least $3/5$ of them are in a ball of radius $r = \epsilon ||A||_F ||B||_F$ with probability $1 - \delta$. To approximate the center of this ball, i.e. A^TB, they use the Las Vegas algorithm. If we only assume that at least $3/5$ of the vectors are in a ball of radius r, approximating the ball this way with probability $1 - \delta$ requires $\Omega((\log \delta^{-1})^2)$ pairwise distance computations and thus $\Omega(d(\log \delta^{-1})^2)$ time where d is the dimension of A^TB as a vector. However, Theorem 2 gives a method that only requires $O(\log \delta^{-1})$ distance computations and $O(d \log \delta^{-1})$ time, thus making amplification of the error for this AMM algorithm linear in $\log \delta^{-1}$.

1-center clustering with outliers is also related to the standard 1-center problem (without outliers), which asks for a point p that minimizes $\max_i \rho(p, a_i)$, where ρ denotes distance [16]. 1-center with outliers has been studied, e.g., in [18], but under the assumption that the number of outliers is $o(n)$, instead of up to $(1 - \alpha)n$. The 1-center and 1-center with outliers problems also have extensions to k-center [3] and k-center with outliers [15, 8], where there

are up to k allowed covering balls. It also relates to the geometric 1-median approximation problem, which asks, for a set of points $a_1, ..., a_n$, for some point p^* such that

$$\sum_{i=1}^{n} \rho(p^*, a_i) \leq C \cdot \min_{p} \sum_{i=1}^{n} \rho(p, a_i),$$

i.e. finding a C-approximation to the geometric 1-median problem. The geometric 1-median problem has been studied in detail, though usually focusing on randomized $(1+\epsilon)$-approximation algorithms in Euclidean space [12, 10]. For the deterministic case in an arbitrary metric space, there exist tight upper [6, 4] and lower time bounds [7, 5] for all C. The geometric 1-median problem is closely related to the 1-center clustering with outliers problem since we will show in Lemma 12 a reduction from geometric 1-median, with slight increases in approximation constant and runtime. Therefore, in combination with the lower bounds of geometric 1-median, this establishes a nontrivial lower bound for 1-center clustering with outliers in general metric space.

As a remark, our Theorem 2 uses an idea of deleting points that are far apart from each other, which is similar to certain ideas for ℓ_1-heavy hitters by Boyer and Moore and by Misra and Gries [2, 17], in which seeing many distinct elements results in a similar deletion process.

1.3 Notation

For many of our proofs, we deal with a weighted generalization of the problem, defined as follows. Let α and $a_1, ..., a_n$ be as in the original problem statement, but now suppose each a_i has some weight $w_i \geq 0$ such that $w_1 + ... + w_n > 0$. Furthermore, assume there is a ball of radius r containing some points $a_{i_1}, ..., a_{i_s}$ such that $w_{i_1} + ... + w_{i_s} \geq \alpha(w_1 + ... + w_n)$. The goal is then to find a ball of radius $O(r)$ containing points $a_{j_1}, ..., a_{j_t}$ such that $w_{j_1} + ... + w_{j_t} \geq \alpha(w_1 + ... + w_n)$, which we call containing at least $\alpha(w_1 + ... + w_n)$ weight.

Given points $a_1, ..., a_n$ with weights $w_1, ..., w_n$, we let $w = \sum_{1 \leq i \leq n} w_i$, i.e. the total weight. For any set $S \subset [n]$, let $a_S = \{a_i : i \in S\}$ and let $w_S = \sum_{i \in S} w_i$. For some results, we define a new set of points $q_1, ..., q_m$ with weights $v_1, ..., v_m$, so we will use the terms "w-weight" and "v-weight" accordingly if necessary. Similarly for any set $S \subset [m]$, let $q_S = \{q_i : i \in S\}$ and let $v_S = \sum_{i \in S} v_i$.

For computing distances, $||x - y||$ denotes distance in a normed vector space, and $\rho(x, y)$ denotes distance in an arbitrary metric space.

Since α, the fraction of points or weight in the ball of radius r, is variable, we define the problem 1-center clustering with approximation constant C and fraction α as the problem where if there is a ball of radius r containing αn points (or αw weight), we wish to explicitly find a ball of radius Cr with the same property.

Finally, for any function f in this paper, the following assumptions are implicit: f is nondecreasing, $f(n) \geq 1$, $f(n) = O(n)$, and $f(an) \leq af(n)$ for any $a \geq 1, n \in \mathbb{N}$.

1.4 Proof Ideas

While many of our proofs assume the weighted problem, we assume the unweighted problem here for simplicity. This is a very minor issue, since the weighted and unweighted problems are almost equivalent, by Lemma 10 (see Appendix A).

The algorithm for the L^p normed vector space simply returns the point whose ith coordinate is the median of the ith coordinates of $a_{[n]}$. The proof is done shortly and is quite brief, so it is not included in this section. We now describe the algorithm intuition for normed vector spaces when $\alpha > \frac{1}{2}$. Our goal is to reduce the n point problem into an $n/2$

point problem in $O_\alpha(nd)$ time, which means the overall runtime is $O_\alpha(nd)$. To do this, we divide the n points into $n/2$ pairs of points just by grouping the first two points, then the next two, and so on. The idea is that when two points are far away, i.e. more than $2r$ apart, at most one of them can actually be in our ball B, so deleting both of them still means at least α of the points are in the ball of radius r. However, when the two points are within $2r$ of each other, we "join" the points by pretending the second point is at the location of the first point, though as a result now we are only guaranteed a ball of radius $3r$ concentric with B having α of the points, because we may join a point in the ball with a point close to the ball but not in it. This means if we have a C approximation for $n/2$ points, we can get a $3C$-approximation for n points, since every remaining pair has the points in the same location so we keep only one point from each pair. However, to go from a ball of radius $3Cr$ to a ball of radius Cr, we look at the original set of points and take the centroid of all the points in the ball of radius $3Cr$. The ball of radius r containing at least αn points will cause the centroid to move closer to the ball, assuming C is not too small. We may have to repeat the process several times with smaller balls until we get sufficiently close, i.e. back to less than Cr away from B, but this only requires $O_\alpha(1)$ iterations and thus $O_\alpha(nd)$ total time.

Unfortunately, for normed vector spaces when $\alpha \leq \frac{1}{2}$, the centroid of the points within a certain radius may not be closer to the desired ball. The idea to fix this is to assume that B has at least αn more points than $B^C \backslash B$ for a certain constant C, where for any A, B^A is the ball of radius Ar concentric with B. Then, if we split the points into two halves, at least one half satisfies the same property. Suppose that given $n/2$ points with this property we can find a ball of radius Kr that not only contains at least αn points but also intersects B, for some $K \leq \frac{C-3}{2}$. Then, the ball of radius $(K+2)r$ around one of these points contains B but is contained in B^C, so if we restrict to the ball of radius $(K+2)r$ around that point, at least $\frac{1+\alpha}{2}$ of the remaining points are in B, which has αn points. Now, use the previous algorithm with some constant which is at least $\frac{1+\alpha}{2} > \frac{1}{2}$ to find a ball of radius Kr with αn points, where we make sure K is not too small. However, there is an issue of multiple completely disjoint balls of radius $O(r)$, each having at least αn points, as $\alpha < \frac{1}{2}$. To salvage this, we have to first find a ball of radius Kr containing αn points, then remove the points in the ball and repeat the procedure with a higher value of α, in case the ball we found does not actually intersect B. Overall, this happens to make the runtime $O(nd \operatorname{polylog} n)$. One issue is that we don't know whether there is some B that contains at least αn more points than $B^C \backslash B$, but if there were some B of radius r that contains at least αn total points, for some $b = O(\log \alpha^{-1})$, B^{C^b} contains at least $\frac{\alpha}{2}n$ more points than $B^{C^{b+1}} \backslash B^{C^b}$, or else the number of points would become too large. Therefore, we attempt the procedure with fraction $\frac{\alpha}{2}$ for radius r, radius Cr, radius $C^2 r$, and so on until $C^{O(\log \alpha^{-1})} r$. Finally, we can go from nd polylog n to $nd \log^{(k)} n$ using a brute force divide and conquer. Namely, if we can solve the problem in time $ndf(n)$, split the points into buckets of size $f(n)$, run the algorithm on each bucket, perhaps with a smaller value of α, and return $O(\frac{n}{f(n)})$ points in time $O(ndf(f(n)))$. If we choose the points well, we get that most of the chosen points will be at most Cr away from our desired ball B, so with a larger constant on the order of C^2, we can run the algorithm on the $O(\frac{n}{f(n)})$ points, which takes $O(nd)$ time. We can repeat the procedure to get $O(ndf^{(k)}(n))$ for any k, though C may become very large.

Our metric space bound ideas are almost identical in the cases of $\alpha > \frac{1}{2}$ and $\alpha \leq \frac{1}{2}$, except for the issue that when $\alpha \leq \frac{1}{2}$, we run into issues of finding a ball of radius Cr with αn points that isn't near the desired ball of radius r and αn points. This issue is fixed by ideas of removing the points in the ball of radius Cr and retrying the algorithm for a larger value of α if necessary. For simplicity we assume $\alpha > \frac{1}{2}$ for the rest of this section.

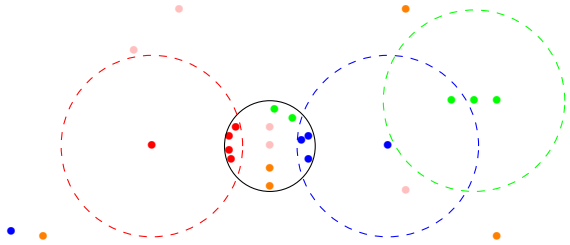


Figure 1 Here is an example for $n = 25, \alpha = 13/25 = 0.52$, and $C = 2$. We split the 25 points into $\sqrt{n} = 5$ buckets of $\sqrt{n} = 5$ points each, color coded red, blue, green, pink, and orange. The black circle represents the desired ball B of radius r. By brute force we try to find a ball of radius $2r$ containing at least an α fraction for each color, and succeed for red, blue, and green (represented by dashed circles). It takes $O(\sqrt{n}^2) = O(n)$ time to try for each color, so the total time for this is $O(n\sqrt{n})$. However, at least an α fraction of points of some color (in this case red) must be in B by Pigeonhole, so the brute force algorithm must succeed in finding a ball of radius $2r$ containing an α fraction of the red points, and since $\alpha > 1/2$, the radius $2r$ ball must contain some point in B and thus must intersect B. This means the ball of radius $4r$ concentric with the dashed red circle must contain B by the triangle inequality, and thus has at least αn points. We can check this for any ball in $O(n)$ time and there are at most $\sqrt{n}$ balls to check, so the total time for this is $O(n\sqrt{n})$.

For metric space upper bounds, one can use brute force divide and conquer. Suppose in time $O(n^{1+1/K})$ we can solve the problem with approximation constant C. Then, split the n points into blocks of size $n^{K/(K+1)}$. If we let the ith block be called D_i, then some block must have at least $\alpha|D_i|$ points. Therefore, if we run the algorithm on all blocks, which takes $O(n \cdot (n^{K/(K+1)})^{1/K}) = O(n^{1+1/(K+1)})$ time, for at least one block we will get a point at most Cr away from B, which means the ball of radius $(C+2)r$ from some point must contain B and thus at least αn total points. There are $O(n^{1/(K+1)})$ points we have to check, each of which takes $O(n)$ time to verify, so we will find a point such that the ball of radius $(C+2)r$ contains at least αn total points in $O(n^{1+1/(K+1)})$ time. As $\alpha > \frac{1}{2}$, this ball by default intersects any ball of radius r with at least αn points. Therefore, if we can solve the problem with approximation constant C in $O(n^{1+1/K})$ time, we can solve the problem with constant $C+2$ in time $O(n^{1+1/(K+1)})$, since the divide and conquer procedure and checking both take $O(n^{1+1/(K+1)})$ time. Since a 2-approximation in n^2 time is trivial, this should give a $2C$ approximation in $O(n^{1+1/C})$ time. See Figure 1 for an example when $C = 2$.

For metric space lower bounds, first it turns out that our divide and conquer technique can be modified to work even with an unknown value for our radius r (see Section 4). It turns out we can also repeat the process $C = \log n$ times rather than a constant number of times to get a $2 \log n$ approximation. The algorithm's time is not quite $n^{1+1/C} = O(n)$ since the constants in the big O get larger: the total time ends up being $O(n \log n)$. An $O(n \log n)$ time solution with a $2 \log n$-approximation helps us bound the minimum value of r by $s \leq r \leq 2s \log n$, where we found a $2s \log n$-radius ball with αn points. Say that p is the geometric median of $a_1, ..., a_n$ and R is the radius of the smallest ball around p with at least αn points. Then, $\sum \rho(p, a_i) \geq (1-\alpha)Rn$ since at least $(1-\alpha)n$ points are at least R away from p. However, if we knew the value of r exactly, then if there is an algorithm that solves metric 1-center clustering with outliers with fraction α and approximation constant C in $O(nf(n))$ time, then the point p^* we get is at most $Cr + R \leq (C+1)R$ away from p, and thus by the triangle inequality $\sum \rho(p^*, a_i) - \sum \rho(p, a_i) \leq (C+1)Rn$. This thereby gets a $\frac{C+1}{1-\alpha} + 1$ approximation to geometric 1-median, but the lower bounds in [5] that deal with geometric 1-median in arbitrary metric spaces show that a $2h - \Theta(1)$-approximation to geometric 1-median requires

$\Omega(n^{1+1/h})$ time. We have not dealt with the fact that we don't exactly know r, but since we have an $O(\log n)$ approximation, we can try $O(\epsilon^{-1} \log\log n)$ attempts of setting r between t and $(1+\epsilon)t$ for $t = s(1+\epsilon)^b$, so the overall time is $O(n \log n + \epsilon^{-1} n f(n) \log\log n)$, which for $f(n) = n^{1+1/K}$ is at most $O(n^{1+1/(K-1)})$. We may be slightly off with our guess for r, but our geometric 1-median approximation only becomes about $1+\epsilon$ times as bad.

2 Normed Vector Space Algorithms: $\alpha > 1/2$

For L^p norms over $\mathbb{R}^d$, there exists a straightforward algorithm. Assume we are given the points $a_1, ..., a_n$ with weights $w_1, ..., w_n$ such that $(a_j)_i$ is the ith coordinate of a_j. Then, consider the point $x = (x_1, ..., x_d)$ such that x_i is the weighted median of $(a_1)_i, ..., (a_n)_i$ where $(a_j)_i$ has weight w_j. Weighted median finding is known to take $O(n)$ time, so x can be found in $O(nd)$ time. Clearly, if there is a ball of radius r around some q with αw weight, where $\alpha > \frac{1}{2}$, then clearly $|q_i - x_i| \leq r$ for each i, so $||q-x||_p \leq r \cdot d^{1/p}$. However, we can actually get another more valuable bound.

▶ **Theorem 1.** *If q is a point such that $B(q)$, the L^p-norm ball of radius r around q, contains αw weight for some $\alpha > \frac{1}{2}$, then $||x-q||_p \leq \left(\frac{\alpha}{\alpha - 1/2}\right)^{1/p} r$, implying an $O(nd)$ time solution with fraction α and approximation constant $O((\alpha - 1/2)^{-1/p})$.*

Proof. Let $(q_1, ..., q_d)$ be the coordinate representation of q, and assume WLOG that $q_i \leq x_i$ for each $1 \leq i \leq d$. Suppose $B(q)$ contains exactly βw weight, where $\beta \geq \alpha$. If we let $(a_j)_i$ denote the ith coordinate of point a_j, the set of points in $\{a_1, ..., a_n\} \cap B(q)$ with $(a_j)_i \geq x_i$ have at least $(\beta - 1/2)w$ weight, as x_i is the weighted median of the ith coordinate of all n points. Therefore,

$$\sum_{a_j \in B(q)} w_j |(a_j)_i - q_i|^p \geq \left(\beta - \frac{1}{2}\right) w (x_i - q_i)^p$$

for each i, meaning that if we sum over all i,

$$\sum_{a_j \in B(q)} w_j ||a_j - q||_p^p = \sum_{i=1}^{d} \sum_{a_j \in B(q)} w_j |(a_j)_i - q_i|^p$$

$$\geq \sum_{i=1}^{d} \left(\beta - \frac{1}{2}\right) w (x_i - q_i)^p = \left(\beta - \frac{1}{2}\right) w ||x-q||_p^p.$$

However, $a_j \in B(q)$ means $||a_j - q||_p^p \leq r^p$, and as the weight of points in $B(q)$ equals βw,

$$(\beta w) \cdot r^p \geq \sum_{a_j \in B(q)} w_j ||a_j - q||_p^p \geq \left(\beta - \frac{1}{2}\right) w ||x-q||_p^p$$

which implies that

$$||x-q||_p \leq \left(\frac{\beta}{\beta - \frac{1}{2}}\right)^{1/p} r \leq \left(\frac{\alpha}{\alpha - \frac{1}{2}}\right)^{1/p} r.$$

Thus, the ball of radius $\left(\left(\frac{\alpha}{\alpha - 1/2}\right)^{1/p} + 1\right) r$ around x contains $B(q)$, and therefore contains at least αw weight. ◀

We next present an algorithm that runs in $O_\alpha(nd)$ time for any normed vector space with fraction $\alpha > \frac{1}{2}$ and approximation constant $O((\alpha - 1/2)^{-1})$, if distances and vector addition/scalar multiplication can be computed in $O(d)$ time, which is true for $\mathbb{R}^d$ with an L^p norm, for example.

▶ **Theorem 2.** *For $\alpha > \frac{1}{2}$, in any normed vector space, if distances and addition/scalar multiplication of vectors can be calculated in $O(d)$ time, there exists an algorithm that solves the weighted problem in $O_\alpha(nd)$ time with fraction α and approximation constant $C = \frac{4\alpha}{2\alpha-1}$.*

Proof. If $n = 1$ we just return the first point so assume $n \geq 2$. Given n points, split the points into $n/2$ groups of 2. Assume n is even, since if n is odd, we can add a final point with 0 weight. Letting $m = \frac{n}{2}$, we construct balls $B_1, ..., B_m$, each of radius $2r$ as follows. The ball B_i will be centered around the point a_{2i-1} or a_{2i} with higher weight (we break ties with a_{2i-1}), so if $w_{2i-1} \geq w_{2i}$ we center around a_{2i-1} and if $w_{2i-1} < w_{2i}$ we center around a_{2i}.

Let q_i be the center of B_i, i.e. q_i is either a_{2i-1} or a_{2i}. Let B be a ball of radius r containing points of total weight at least αw, and let q be the center of B.

We construct the new set of weights v_i for the points q_i. We let v_i be the total w-weight of the subset of $\{a_{2i-1}, a_{2i}\}$ which is contained in B_i minus the total w-weight of the subset which is not contained in B_i. In other words, if $||a_{2i-1} - a_{2i}|| \leq 2r$, then $v_i = w_{2i-1} + w_{2i}$ and otherwise, $v_i = \max(w_{2i-1}, w_{2i}) - \min(w_{2i-1}, w_{2i})$. Note that the total weight of $\{a_{2i-1}, a_{2i}\} \cap B_i$ is $\frac{w_{2i-1}+w_{2i}+v_i}{2}$. Clearly, for all i, $0 \leq v_i \leq w_{2i-1} + w_{2i}$.

Next, if $||q_i - q|| > 3r$, then B_i and B do not intersect. This means that the total w-weight of $\{a_{2i-1}, a_{2i}\} \cap B$ is at most $\frac{w_{2i-1}+w_{2i}-v_i}{2}$. If $||q_i - q|| \leq 3r$, the total w-weight of the intersection $\{a_{2i-1}, a_{2i}\} \cap B$ is at most $\frac{w_{2i-1}+w_{2i}+v_i}{2}$, since if both $a_{2i-1}, a_{2i} \in B$, then both are in B_i, and if exactly one of a_{2i-1}, a_{2i} is in B, then the one with larger weight is in B_i because it is the center, q_i.

Now, define $S \subset [m]$ to be the set of i such that $||q_i - q|| \leq 3r$, i.e. $S = \{i : 1 \leq i \leq m, ||q_i - q|| \leq 3r\}$. Then, by looking at the total w-weight of the subset of $a_{[n]}$ in B,

$$\sum_{i \in S} \frac{w_{2i-1} + w_{2i} + v_i}{2} + \sum_{i \notin S} \frac{w_{2i-1} + w_{2i} - v_i}{2} \geq \sum_{a_i \in B} w_i \geq \alpha w.$$

Since w is nonzero and $\alpha > \frac{1}{2}$, at least one v_i is nonzero. The left hand side equals

$$\frac{w}{2} + \frac{1}{2} \sum_{i \in S} v_i - \frac{1}{2} \sum_{i \notin S} v_i,$$

which means

$$\sum_{i \in S} v_i - \sum_{i \notin S} v_i \geq (2\alpha - 1)w \geq (2\alpha - 1) \sum_{1 \leq i \leq m} v_i \Rightarrow \sum_{i \in S} v_i \geq \alpha \sum_{1 \leq i \leq m} v_i.$$

Therefore, the ball of radius $3r$ around q contains at least α of the total v-weight of the points q_i. Since at least one of the v_i's is nonzero and all are nonnegative, we can find a ball of radius $3Cr$ around some point p containing at least α of the total v-weight by performing the same algorithm on a size m set $q_1, ..., q_m$. Therefore, the ball of radius $3r$ around q intersects the ball of radius $3Cr$ around p, as some q_i must be in both balls, so the ball of radius $(3C + 4)r$ around p must contain B. Given this, if we can get some ball of radius Cr that contains B, we are done.

We do this via looking at centroids, where the weighted centroid of points $x_1, ..., x_m$ with weights $w_1, ..., w_m$ equals $\frac{w_1 x_1 + ... + w_m x_m}{w_1 + ... + w_m}$. Let $\epsilon = \alpha - \frac{1}{2}$ and choose some $K \geq 2 + \frac{1}{\epsilon}$.

Suppose we have found some point a such that the ball of radius Kr around a, denoted $B^K(a)$, contains B. We look at the w-weighted centroid of all points $a_i \in B^K(a)$, which clearly takes $O(nd)$ time to calculate. If we let $a_{S_1} = a_{[n]} \cap B$, then $w_{S_1} \geq \alpha w$ so the sum of the w-weights of points in $B^K(a) \backslash B$ is at most $w(1-\alpha)$. Then, the distance between the weighted centroid of all $a_i \in B^K(a)$ and q is at most

$$\frac{1}{w_{S_1} + \sum_{a_i \in B^K(a)\backslash B} w_i} \left(\sum_{a_i \in B} ||q - a_i|| w_i + \sum_{a_i \in B^K(a)\backslash B} ||q - a_i|| w_i \right)$$

$$\leq \frac{1}{w_{S_1} + \sum_{a_i \in B^K(a)\backslash B} w_i} \left(r w_{S_1} + (2K-1)r \sum_{a_i \in B^K(a)\backslash B} w_i \right)$$

since $||q - a|| \leq (K-1)r$ and $||a - a_i|| \leq Kr$ for any $a_i \in B^K(a) \backslash B$. But since $w_{S_1} \geq \alpha w$ and $\sum_{a_i \in B^K(a)\backslash B} w_i \leq (1-\alpha)w$, this is at most

$$\alpha r + (2K-1)(1-\alpha)r = (2K - 1 - 2K\alpha + 2\alpha)r = (2K - 1 - K - 2K\epsilon + 1 + 2\epsilon)r = (K - 2(K-1)\epsilon)r.$$

However, since $K \geq 2 + \frac{1}{\epsilon}$, $2(K-1)\epsilon \geq K\epsilon + 1$, so this is at most $(K - K\epsilon - 1)r$. Therefore, the weighted centroid of all these points is at most $(K - K\epsilon - 1)r$, so the ball of radius $K(1-\epsilon)r$ around the weighted centroid contains B. This gives us a slightly better range. We can repeat this process starting with $K = 3C + 4$ until we get $K \leq C$, assuming that $C = 2 + \frac{1}{\epsilon} = \frac{4\alpha}{2\alpha - 1}$. As $3C + 4 \leq 5C$, this process needs to be repeated at most $(\log 5)/(\log \frac{1}{1-\epsilon}) = O(\epsilon^{-1})$ times.

With the exception of the recursion on $q_1, ..., q_m$ with weights $v_1, ..., v_m$, everything else takes $O(nd)$ time, but we have to repeat the centroid algorithm multiple times, where the number of repetitions depends on α. Therefore, the total running time is $T(n) = O_\alpha(nd) + T(n/2)$, which means $T(n) = O_\alpha(nd)$, as desired. ◀

3 Normed Vector Space Algorithms: $\alpha > 0$

While we were unable to solve the normed vector space 1-center clustering with outliers problem for all $\alpha > 0$ in $O_\alpha(nd)$ time, we were able to find a solution running in $O_{\alpha,k}(nd \log^{(k)} n) = O_{\alpha,k}(nd \log ... \log(n))$ time. We first show an nd polylog n time solution and explain how this can be used to solve the problem in $O_{\alpha,k}(nd \log^{(k)} n)$ time.

The following result is useful for both the normed vector space and arbitrary metric space versions, primarily for $0 < \alpha \leq \frac{1}{2}$. It is important for making sure that if we found a ball of radius Cr containing αw weight or αn points, even if there are multiple disjoint balls with this property, we can find a few balls of radius Cr, of which any ball of radius r containing at least αw weight or αn points is near one of the radius Cr balls.

▶ **Lemma 3.** *Suppose we are in some space where computing distances between two points can be done in $O(d)$ time. Suppose that for some fixed α, C and for any $\beta \geq \alpha$, we can solve the weighted problem with fraction β and approximation constant C in time $O(ndf(n))$ (with the runtime constant independent of β). Then, for any $\beta \geq \alpha$, we can find at most β^{-1} points $p_1, ..., p_\ell$ in $O(ndf(n)\lfloor \beta^{-1} \rfloor)$ time such that the ball of radius Cr around each p_i contains at least βw weight and any ball of radius r containing at least βw total weight intersects at least one of the balls of radius Cr.*

The proof of lemma 3 is not too difficult and is left in Appendix B.

▶ **Lemma 4.** *For any $0 < \alpha < 1$, let $C = 2 + \frac{2}{\alpha}$ and assume we are dealing with the weighted problem in a normed vector space (with $w > 0$), where distances and vector addition/scalar multiplication are calculable in $O(d)$ time. Suppose there exists a ball B of radius r such that B and the ball B^{2C+3} concentric with B but of radius $(2C+3)r$ satisfies*

$$\sum_{a_i \in B} w_i \geq \left(\sum_{a_i \in B^{2C+3} \setminus B} w_i \right) + \alpha w.$$

Then, we will be able to find a set of at most $\frac{1}{\alpha}$ points $z_1, ..., z_\ell$ in $O_\alpha(nd(\log n)^{\lfloor \alpha^{-1} \rfloor})$ time such that the ball $B^C(z_i)$ of radius Cr around each z_i contains at least αw total weight, and at least one of the balls $B^C(z_i)$ intersects the ball B.

Also, if there does not exist such a ball B, the algorithm will still succeed and satisfy the conditions (where the condition of B intersecting at least one of $B^C(z_i)$ is true by default).

Proof. Our proof inducts on $\lfloor \alpha^{-1} \rfloor$. We show an $O(nd \log n)$-time algorithm for $\alpha > \frac{1}{2}$ and given an $O(nd(\log n)^{k-1})$-time algorithm for all $\alpha' > \frac{1}{k}$, we show an $O(nd(\log n)^k)$-time algorithm for all $\alpha > \frac{1}{k+1}$. This means that the big O time constant may depend on $\lfloor \alpha^{-1} \rfloor$.

Assume n is a power of 2, as we can add extra points of weight 0. Next, split up the points $a_1, ..., a_n$ into two groups $a_{[n/2]}$ and $a_{[n/2+1::n]}$. Note that B clearly still holds the same property for either the first half or second half of points, i.e. either

$$\sum_{\substack{a_i \in B \\ 1 \leq i \leq n/2}} w_i \geq \alpha w_{[n/2]} + \sum_{\substack{a_i \in B^{2C+3} \setminus B \\ 1 \leq i \leq n/2}} w_i \quad \text{or} \quad \sum_{\substack{a_i \in B \\ n/2+1 \leq i \leq n}} w_i \geq \alpha w_{[n/2+1::n]} + \sum_{\substack{a_i \in B^{2C+3} \setminus B \\ n/2+1 \leq i \leq n}} w_i.$$

The algorithm first recursively runs on the two halves $a_{[n/2]}$ and $a_{[n/2+1::n]}$ to get points $x_1, ..., x_r$ and $y_1, ..., y_s$ such that $r, s \leq \frac{1}{\alpha}$ and there exists some point $z \in \{x_1, ..., x_r, y_1, ..., y_s\}$ such that the ball of radius Cr around z intersects B. Therefore, $B^{C+2}(z)$, the ball of radius $(C+2)r$ around z, contains B but is contained in B^{2C+3}.

Suppose we could successfully guess such a point z. Then, the weight of points in $a_{[n]} \cap B^{C+2}(z)$ is βw for some $\beta \geq \alpha$, and so the weight of points in $a_{[n]} \cap B$ is at least $\frac{\beta+\alpha}{2} w$ since $B^{C+2}(z) \subset B^{2C+3}$. We can easily determine the set of points in $a_{[n]} \cap B^{C+2}(z)$ in $O(nd)$ time, and thus compute β. Now, among the points in $a_{[n]} \cap B^{C+2}(z)$, at least $\frac{\beta+\alpha}{2\beta} \geq \frac{1+\alpha}{2}$ of the weight is contained in some ball of radius r, which means by Theorem 2, we can in $O_\alpha(nd)$ time find a ball of radius

$$\frac{4\left(\frac{\beta+\alpha}{2\beta}\right)}{2\left(\frac{\beta+\alpha}{2\beta}\right) - 1} \cdot r = \frac{2(\beta+\alpha)}{\beta+\alpha-\beta} \cdot r = \left(2 + \frac{2\beta}{\alpha}\right) r \leq Cr$$

containing at least $\frac{\beta+\alpha}{2\beta} \cdot \beta w \geq \alpha w$ weight.

If $\alpha > \frac{1}{2}$, this means we have found a ball of radius Cr with at least αw total weight. It must also intersect B, because otherwise the total weight of all the points would be at least $2\alpha w > w$. Therefore, we can recursively run the algorithm on the two halves, and then in $O(nd)$ time guess at most 2 possibilities for z to find a ball of radius Cr. Therefore, this algorithm takes $T(n) = 2T(n/2) + O(nd) \Rightarrow T(n) = O(nd \log n)$ time.

Suppose $\frac{1}{k+1} < \alpha \leq \frac{1}{k}$. Then, in $O_\alpha(nd)$ time, we can try each $z \in \{x_1, ..., y_s\}$ to get some ball of radius Cr centered around $z_1 = z$ that contains at least αw weight. If we find no such ball, then no such B exists, so we return nothing. Else, we find some ball around z_1. In case the ball does not intersect B, we compute the total weight of points in $B^C(z_1)$,

the ball of radius Cr around z_1. Define γ so that the weight of points in $B^C(z_1)$ equals γw, so clearly $\gamma \geq \alpha$. Therefore, if $B^C(z_1)$ does not intersect B, then if we remove these points, we have a subset $\{a'_1, ..., a'_m\}$ of the original points with total weight $w' = (1-\gamma)w$, which means that for the new set of points, B satisfies

$$\sum_{a'_i \in B} w'_i = \sum_{a_i \in B} w_i \geq \left(\sum_{a_i \in B^{2C+3} \setminus B} w_i \right) + \alpha w \geq \left(\sum_{a'_i \in B^{2C+3} \setminus B} w'_i \right) + \frac{\alpha}{1-\gamma} w'.$$

Thus, by our induction hypothesis, in $O_{\alpha/(1-\gamma)}(nd(\log n)^{\lfloor (1-\gamma)/\alpha \rfloor}) = O_\alpha(nd(\log n)^{\lfloor \alpha^{-1} \rfloor - 1})$ time, we can find a set of at most $\frac{1-\gamma}{\alpha} \leq \frac{1}{\alpha} - 1$ points $z_2, ..., z_\ell$ such that the balls of radius Cr around each z_i contains at least $\frac{\alpha}{1-\gamma} w' = \alpha w$ weight in the new set of points (and thus in the old set of points), and at least one of the balls of radius Cr around some z_i (possibly z_1) intersects B.

Since we first recursively perform the algorithm on the two halves, the total runtime is $T(n) = 2 \cdot T(n/2) + O_\alpha(nd(\log n)^{\lfloor \alpha^{-1} \rfloor - 1})$ by our inductive hypothesis, so $T(n) = O_\alpha(nd(\log n)^{\lfloor \alpha^{-1} \rfloor})$. ◀

We use the previous result to find an $O(nd \text{ polylog } n)$ time solution.

▶ **Lemma 5.** *For any $0 < \alpha < 1$, one can solve the weighted Euclidean problem with fraction α and some approximation constant $C = O_\alpha(1)$ in $O_\alpha(nd(\log n)^{\lfloor 2\alpha^{-1} \rfloor})$ time.*

Proof. Suppose B is a ball of radius r around p with αw points and let $S \subset \mathbb{N} \cup \{0\}$ be the set of nonnegative integers s such that there is a ball of radius $\left(\frac{8}{\alpha} + 7\right)^s \cdot r$ around p containing at least $(\frac{3}{2})^s \cdot \alpha w$ total weight. Because of B, $0 \in S$. Since $\alpha > 0$, there clearly exists a maximal $s \in S$ which is at most $\frac{\log(\alpha^{-1})}{\log(3/2)}$. For this maximal s, there is a ball B' of radius $R = \left(\frac{8}{\alpha} + 7\right)^s \cdot r$ around p containing at least $\alpha' w$ weight, where $\alpha' = (\frac{3}{2})^s \alpha$, but the ball of radius $\left(\frac{8}{\alpha} + 7\right) R$ around p contains at most $\frac{3}{2}\alpha' w$ total weight. Therefore, if $\beta = \frac{\alpha}{2}$, if we let $C = 2 + \frac{2}{\beta}$, the ball $(B')^{2C+3}$ of radius $(2C+3)R = \left(\frac{8}{\alpha} + 7\right) R$ around p satisfies

$$\sum_{a_i \in B'} w_i \geq \left(\sum_{a_i \in (B')^{2C+3} \setminus B'} w_i \right) + \beta w.$$

Therefore, if we knew s, plugging β into the algorithm of Lemma 4 gives us, in $O_\alpha(nd(\log n^{\lfloor 2\alpha^{-1} \rfloor}))$ time, at most $2\alpha^{-1}$ points such that the ball of radius $\left(\frac{4}{\alpha} + 2\right) \cdot \left(\frac{8}{\alpha} + 7\right)^s$ around at least one of them intersects B', and thus the ball of radius $\left(\frac{4}{\alpha} + 4\right) \cdot \left(\frac{8}{\alpha} + 7\right)^s$ around that point has at least αw weight. We can try it for all s between 0 and $\frac{\log(\alpha^{-1})}{\log(3/2)}$ and verify each point (verification takes $O_\alpha(nd)$ time) to get at least one ball containing αw or more weight, which gives the desired result. ◀

We now can go to $O_{\alpha,k}(nd \log^{(k)}(n))$ time using the following lemma.

▶ **Lemma 6.** *Fix some α, C and suppose we are in some space (Euclidean, general metric, or something else) where distances can be computed in $O(d)$ time. Suppose that for any fraction $\beta \geq \alpha$ and approximation constant C there exists an algorithm that solves the weighted problem in time $O(ndf(n))$. Then, for any nondecreasing function $g(n)$ such that $1 \leq g(n) \leq n$, there is an algorithm that runs in $O\left(ndf(g(n)) + \frac{ndf(n)}{g(n)}\right)$ with fraction $\alpha' = \sqrt{2\alpha}$ and approximation constant $C' = C^2 + 2C + 2$.*

Proof. We use a similar divide and conquer approach to Lemma 4. Partition $[n]$ into buckets $D_1, ..., D_m$, each of size $\Theta(g(n))$, which gives us a partition of points $a_{D_1}, ..., a_{D_m}$. If B is a ball of radius r containing at least $\alpha' w$ total weight, then let v_i be the total weight of all points in $a_{D_i} \cap B$. If $S \subset [m]$ is the set of all i such that $v_i > \frac{\alpha'}{2} w_{D_i}$, then

$$\alpha' w \leq \sum_{a_j \in B} w_j = \sum_{i \in [m]} \sum_{\substack{j \in D_i \\ a_j \in B}} w_j \leq \sum_{i \in S} w_{D_i} + \frac{\alpha'}{2} \sum_{i \notin S} w_{D_i} \leq \frac{\alpha' w}{2} + \sum_{i \in S} w_{D_i},$$

and thus $\frac{\alpha'}{2} w \leq \sum_{i \in S} w_{D_i}$.

For each $1 \leq i \leq m$, by Lemma 3, since $\alpha' \geq \alpha$, there is an $O(ndf(g(n)))$-time algorithm which returns for each $i \in [m]$ at most α'^{-1} points $p_{i,1}, ..., p_{i,\ell_i}$ such that if $i \in S$, the ball of radius Cr around at least one of the points intersects B. Therefore, for every $i \in S$, some $p_{i,j}$ is at most $(C+1)r$ from the center of B. Now, we can compute $w_{D_1}, ..., w_{D_m}$ in $O(n)$ time and assign each $p_{i,j}$ weight w_{D_i}. Then, the total weight of all $p_{i,j}$ is at most $\alpha'^{-1} w$. However, for an individual $i \in S$, the total weight of the points $p_{i,j}$ for all $1 \leq j \leq \ell_i$ in the ball of radius $(C+1)r$ around B is at least w_{D_i} since at least one $p_{i,j}$ is in the ball. Therefore, the total weight of all points $p_{i,j}$ in the ball of radius $(C+1)r$ around B is at least $\sum_{i \in S} w_{D_i} \geq \frac{\alpha'}{2} w$, which is at least $\frac{\alpha'^2}{2}$ times the total weight of all the $p_{i,j}$'s. Therefore, by Lemma 3, applying the algorithm for $\alpha = \frac{\alpha'^2}{2}$ on the $p_{i,j}$'s with the new radius $(C+1)r$ gives a set of at most α^{-1} points $q_1, ..., q_\ell$ such that the ball of radius $C(C+1)r$ around at least one of the q_i's intersects the ball of radius $(C+1)r$ around the center of B. This algorithm takes $O(\alpha^{-1} mdf(m)) = O(nd\frac{f(n)}{g(n)})$ time, as α is fixed. Therefore, the ball of radius $(C^2+2C+2)r = C'r$ around at least one of the q_i's contains B, so we verify for each q_i if the ball of radius $(C^2+2C+2)r$ contains at least αw total weight, which takes $O(nd)$ time. ◀

▶ **Theorem 7.** *For $\alpha \leq \frac{1}{2}$, the 1-center clustering with outliers problem can be solved in $O_\alpha(nd \log^{(k)}(n))$ time in any normed vector space for some constant $C = O_\alpha(1)$.*

Proof. Letting $f = g$ in Lemma 6 tells us there is an $O(ndf(f(n))$-time algorithm with fraction $\sqrt{2\alpha}$ and approximation constant C^2+2C+2 given an $O(ndf(n))$-time algorithm with fraction α and approximation constant C. Repeating this k times gives us an $O_k(ndf^{(2^k)}(n))$-time algorithm with fraction $2 \cdot (\alpha/2)^{2^{-k}}$ and approximation constant $O_{C,k}(1)$. Therefore, since we have an algorithm running in $O_\alpha(ndf(n))$ with $f(n) = (\log n)^{\lfloor 2\alpha^{-1} \rfloor}$ with approximation constant $O_\alpha(1)$ and fraction α, we have an algorithm that runs in $O_{\alpha,k}(ndf^{(2^k)}(n)) = O_{\alpha,k}(nd \log^{(2^k-1)} n)$ time, with fraction $2 \cdot (\alpha/2)^{2^{-k}}$ and approximation constant $O_{\alpha,k}(1)$. Letting $\beta = 2 \cdot (\alpha/2)^{2^{-k}}$, then $\alpha = (\beta/2)^{2^k}/2$, which means for any $0 < \beta < 1$, there is an $O_{\beta,k}(nd \log^{(2^k-1)}(n))$ time solution with approximation constant $O_{\beta,k}(1)$ and fraction β. ◀

4 Metric Space Upper Bounds

The idea for proving that there is an $O_{\alpha,C}(n^{1+1/C})$-time algorithm with fraction α and approximation constant $2C$ uses induction on $\lfloor \alpha^{-1} \rfloor$ and C. The base case proofs of $\alpha > \frac{1}{2}$ and $C = 1$ are quite similar to the induction step, so we leave their proofs in Appendix B.

▶ **Theorem 8.** *For any $\alpha > 0$, say we are trying to solve weighted 1-center clustering with outliers in a general metric space, where r is unknown. For all $C \in \mathbb{N}$, we can find a set of points $p_1, ..., p_\ell$ and corresponding radii $s_1, ..., s_\ell$, where $\ell \leq \lfloor \alpha^{-1} \rfloor$, such that the ball of radius s_i around p_i contains at least αw of the weight in $O((2\binom{\lfloor \alpha^{-1} \rfloor + C}{C} - \lfloor \alpha^{-1} \rfloor - 1) n^{1+1/C})$ time, assuming $n = m^C$ for some integer m. Moreover, any ball of radius r containing at least αw weight intersects at least one ball of radius s_i around some p_i, for some $s_i \leq 2Cr$.*

Proof. We induct on $\lfloor \alpha^{-1} \rfloor$ and C. The base cases $\lfloor \alpha^{-1} \rfloor = 1$ and $C = 1$ are done in Appendix B. Suppose the theorem holds for all $\alpha' > \frac{1}{z}$ and we are looking at some $\frac{1}{z+1} < \alpha \leq \frac{1}{z}$. Also, suppose we have an algorithm for α and $C-1$.

Split the points into blocks $D_1, ..., D_m$ each of size m^{C-1}. For each block D_i, by our inductive hypothesis we can return points $p_{i,1}, ..., p_{i,\ell_i}$ and radii $r_{i_1}, ..., r_{i,\ell_i} \in a_{D_i}$ where $\ell_i \leq z$ for all i, subject to some conditions. First, the ball $B_{i,k}$ of radius $r_{i,k}$ around $p_{i,k}$ has at least αw_{D_i} weight. Second, if there is a ball of radius r that contains at least αw_{D_i} weight when intersected with a_{D_i}, then the ball must intersect $B_{i,k}$ for some k where $p_{i,k} \leq 2(C-1)r$. Moreover, by our induction hypothesis we can determine these points in time

$$\left(\left(2\binom{z+C-1}{C-1} - z - 1\right)\left(\frac{n}{m}\right)^{1+1/(C-1)} \cdot m\right) = O\left(\left(2\binom{z+C-1}{C-1} - z - 1\right) n^{1+1/C}\right).$$

If B is a ball of radius r containing at least αw total weight, then there exists some $1 \leq j \leq m$ such that $w_{D_j} > 0$ and the total weight of $a_{D_j} \cap B$ is at least αw_{D_j}. Therefore, $B_{j,k}$ intersects B for some $r_{j,k} \leq 2(C-1)r$, so the ball of radius $2Cr$ around $p_{j,k}$ for some j, k when intersected with $a_{[n]}$ contains at least αw total weight. We can check all the $p_{j,k}$ and since weighted median can be solved in $O(n)$ time, we can find some $p_{j,k}$ with the smallest radius $s_{j,k}$ (not necessarily the same as $r_{j,k}$) containing at least αw weight in $O(mz \cdot n) = O(zn^{1+1/C})$. We know that $s_{j,k} \leq 2Cr$, and we can set $p_1 = p_{j,k}$ and $s_1 = s_{j,k}$.

Now, remove every point in the ball of radius s_1 around p_1 by changing their weights to 0. If the total weight of removed points is βw where $\beta \geq \alpha$, the total weight is now $(1-\beta)w$. If there is still some ball of radius r that contains at least αw weight now, then it contains at least $\frac{\alpha}{1-\beta} > \frac{1}{z-1}$ of the total weight now. Therefore, we can use induction on z with $\alpha' = \frac{\alpha}{1-\beta}$. This gives us at most z points $p_1, ..., p_\ell$ and radii $s_1, ..., s_\ell$, where the first point p_1 is our original $p_{j,k}$ and the next $\ell - 1$ points and radii are found in $O\left(\left(2\binom{z-1+C}{C} - (z-1) - 1\right) n^{1+1/C}\right)$ time. Moreover, any ball B of radius r either intersects the ball of radius s_1 around p_1, where $s_1 \leq 2Cr$, or by the induction hypothesis on $\lfloor \alpha^{-1} \rfloor$ intersects some s_i around p_i for some $2 \leq i \leq \ell$ with $s_i \leq 2Cr$, since B would have at least $\frac{\alpha}{1-\beta}$ of the remaining weight if it doesn't intersect the ball of radius s_1 around p_1.

Therefore, the total time is

$$O\left(\left(2\binom{z+C-1}{C-1} - z - 1\right) n^{1+1/C} + \left(2\binom{z-1+C}{C} - z\right) n^{1+1/C} + zn^{1+1/C}\right)$$

$$= O\left(\left(2\binom{z+C}{C} - z - 1\right) n^{1+1/C}\right).$$

◀

▶ Remark. C does not have to be a constant independent of n, since the O factor is independent of z and C. For example, $m = 2, C = \lceil \lg n \rceil$, the theorem still holds.

As we can add points of 0 weight until we get a perfect power of C, we have the following.

▶ **Corollary 9.** *In any metric space, we can find a ball of radius $2Cr$ with at least αn points in $O_{\alpha,C}(n^{1+1/C})$ time, given that there exists a ball of radius r with at least αn points.*

References

1 Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. *J. Comput. Syst. Sci.*, 58(1):137–147, 1999.

2 Robert S. Boyer and J. Strother Moore. MJRTY—A Fast Majority Vote Algorithm. In *Automated Reasoning: Essays in Honor of Woody Bledsoe*, pages 105–117. Springer Netherlands, 1991.

3 Deeparnab Chakrabarty, Prachi Goyal, and Ravishankar Krishnaswamy. The Non-Uniform k-Center Problem. In *43rd International Colloquium on Automata, Languages, and Programming*, volume 55 of *ICALP '16*, pages 67:1–67:15, 2016.

4 Ching-Lueh Chang. Deterministic sublinear-time approximations for metric 1- median selection. *Information Processing Letters*, 113:288–292, 2013.

5 Ching-Lueh Chang. A lower bound for metric 1-median selection. *Journal of Computer and System Sciences*, 84, 2014.

6 Ching-Lueh Chang. A deterministic sublinear-time nonadaptive algorithm for metric 1-median selection. *Theoretical Computer Science*, 602:149–157, 2015.

7 Ching-Lueh Chang. Metric 1-median selection: Query complexity vs. approximation ratio. *Transactions on Computation Theory*, 9:20:1–20:23, 2018.

8 Moses Charikar, Samir Khuller, David M. Mount, and Giri Narasimhan. Algorithms for facility location problems with outliers. In *SODA '01*, pages 642–651, 2001.

9 Moses Charikar, Jacob Steinhardt, and Gregory Valiant. Learning from untrusted data. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, pages 47–60, 2017.

10 Hui Han Chin, Aleksander Madry, Gary L. Miller, and Richard Peng. Runtime guarantees for regression problems. In *ITCS*, pages 269–282, 2013.

11 Kenneth L. Clarkson and David P. Woodruff. Numerical linear algebra in the streaming model. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, STOC '09, pages 205–214, 2009.

12 Michael B. Cohen, Yin Tat Lee, Gary Miller, Jakub Pachocki, and Aaron Sidford. Geometric median in nearly linear time. In *Proceedings of the Forty-eighth Annual ACM Symposium on Theory of Computing*, STOC '16, pages 9–21, 2016.

13 I. Diakonikolas, G. Kamath, D. Kane, J. Li, A. Moitra, and A. Stewart. Robust estimators in high dimensions without the computational intractability. In *Proceedings of the 57th Annual IEEE Symposium on Foundations of Computer Science*, FOCS 2016, 2016.

14 Peter J. Huber and Elvezio Ronchetti. *Robust Statistics*. Wiley, 2009.

15 R. M. McCutchen and S. Khuller. Streaming algorithms for k-center clustering with outliers and with anonymity. In *APPROX-RANDOM*, pages 165–178, 2008.

16 Nimrod Megiddo. The weighted euclidean 1-center problem. *Mathematics of Operations Research*, 8(4):498–504, 1983.

17 J. Misra and David Gries. Finding repeated elements. *Science of Computer Programming*, 2:143–152, 1982.

18 Hamid Zarrabi-Zadeh and Asish Mukhopadhyay. Streaming 1-center with outliers in high dimensions. In *Proceedings of the 21st Annual Canadian Conference on Computational Geometry*, CCCG '09, pages 83–86, 2009.

A Metric Space Lower Bounds

The following lemma is useful for showing the intuitive fact that higher values α make the problem easier. The lemma also shows that the weighted and unweighted problems are almost equivalent. As a result, it isn't too important whether we are dealing with the weighted or unweighted problem for our bounds. In conjunction with Lemma 3, it establishes that given an algorithm solving the 1-center clustering with outliers problem for some α, we can efficiently find a few disjoint balls of radius $O(r)$ such that any ball of radius r containing at least αw weight or αn points is near one of the radius Cr balls. We require this for a similar reason as we needed it for the normed vector space upper bounds for $\alpha > \frac{1}{2}$.

▶ **Lemma 10.** *Suppose we are in some space (Euclidean, general metric, or something else) where computing distances between two points can be done in $O(d)$ time. Suppose that for some fixed α, C, we can solve the unweighted problem with fraction α and approximation constant C in time $O(ndf(n))$. Then, for any $\beta > \alpha$, we can solve the weighted problem with fraction β and approximation constant $C+2$ in time $O((\beta-\alpha)^{-2}\alpha^{-2}\log\alpha^{-1}ndf(n))$.*

Proof. Suppose that the points are $a_1, ..., a_n$ and the weights are $w_1, ..., w_n$. Let $\overline{w}$ be the average of the weights, i.e. $\frac{w}{n}$. Now, for $\epsilon = \beta - \alpha$, define $\overline{w}_i$ as $\epsilon\overline{w}\lfloor\frac{w_i}{\epsilon\overline{w}}\rfloor$, i.e. w_i rounded down to the nearest multiple of $\epsilon\overline{w}$. Then, $\overline{w}_1 + ... + \overline{w}_n \leq w_1 + ... + w_n$, but if there exists a ball B of radius r such that

$$\sum_{a_i \in B} w_i \geq \beta \sum_{i=1}^{n} w_i = \beta w,$$

then

$$\sum_{a_i \in B} \overline{w}_i \geq \sum_{a_i \in B} (w_i - \epsilon\overline{w}) \geq \beta w - \epsilon \cdot n \cdot \overline{w} = \alpha w \Rightarrow \sum_{a_i \in B} \overline{w}_i \geq \alpha \sum_{i=1}^{n} \overline{w}_i.$$

However, note that

$$\sum_{i=1}^{n} \overline{w}_i \leq \sum_{i=1}^{n} w_i = n\overline{w} = (\epsilon\overline{w})\frac{1}{\epsilon} \cdot n,$$

which means if we consider the unweighted problem with each point a_i repeated $\frac{\overline{w}_i}{\epsilon\overline{w}}$ times, we have at most $\frac{1}{\epsilon} \cdot n$ points, of which at least an α fraction of them are in the ball B. Therefore, we use the algorithm that solves the unweighted problem in time $O(\frac{n}{\epsilon}df(\frac{n}{\epsilon})) = O(\epsilon^{-2}ndf(n))$ time to find some ball of radius Cr around some point p which contains at least an α fraction of these points.

Note that this ball we found by solving the unweighted problem, which we denote by B^C, has at least α of the new $\overline{w}$-weight, which means

$$\sum_{a_i \in B^C} w_i \geq \sum_{a_i \in B^C} \overline{w}_i \geq \alpha \sum_{i=1}^{n} \overline{w_i} \geq \alpha(w - n \cdot \epsilon\overline{w}) = \alpha(1-\epsilon)w.$$

Next, we check if B^{C+2} contains at least βw total weight. If so we are done, and if not we remove all points inside B^C. Since B^{C+2} doesn't contain βw weight, it doesn't contain B and thus B^C and B are disjoint. If B^C contains $\alpha' w \geq \alpha(1-\epsilon)w$ weight, then the set of remaining points has $(1-\alpha')w$ weight, so to find a ball of radius $(C+2)r$ with βw total weight, we have to solve the weighted problem for the new set and fraction $\frac{\beta}{1-\alpha'}$. Since

$\frac{\beta}{1-\alpha'} \geq \frac{\beta}{1-\alpha(1-\epsilon)} > \frac{\beta}{1-\alpha^2}$, to solve the problem for β, it suffices to solve the problem for $\frac{\beta}{1-\alpha'}$. We repeat this process with a larger value of β each time, and as β multiplies by at least $(1-\alpha^2)^{-1}$ each time, we need at most $O(\alpha^{-2}\log\alpha^{-1})$ times to reduce it to solving for some $\beta \geq 1$, by which time the problem has become trivial.

The process each time takes $O((\beta'-\alpha)^{-2}ndf(n)) = O((\beta-\alpha)^{-2}ndf(n)$ time for each iteration, where β' is the fraction at the current iteration. This gives us the desired runtime. ◀

We next note a corollary of Theorem 8 that is actually useful for proving lower bounds. This is because it helps us reduce from the Geometric 1-Median approximation problem.

▶ **Lemma 11.** *Given $a_1, ..., a_n$ in a general metric space and some $0 < \alpha < 1$, we can return at most $\lfloor \alpha^{-1} \rfloor$ points $p_1, ..., p_\ell$ with radii $r_1, ..., r_\ell$ in $O(n(\lceil \log n \rceil)^{\lfloor \alpha^{-1} \rfloor})$ time such that if there is a ball B of radius r covering more than αn points, then for some i, $r_i \leq 2\lceil \lg n \rceil r$ and the ball of radius p_i around r_i intersects B.*

Proof. The proof follows from Theorem 8. Assume n is a power of 2 by adding some extra points of weight 0. Then, let $C = \lg n$. Since $n^{1/\lg n} = 2$ and $\binom{\lfloor \alpha^{-1} \rfloor + \lg n}{\lg n} \leq (\lg n)^{\lfloor \alpha^{-1} \rfloor}$, $\binom{\lfloor \alpha^{-1} \rfloor + \lg n}{\lg n} \cdot n \cdot n^{1/\lg n} = O(n\lceil \lg n \rceil^{\lfloor \alpha^{-1} \rfloor})$. Therefore, we can directly apply Theorem 8. ◀

▶ **Lemma 12.** *Fix some $0 < \alpha < 1$ and C. Suppose in $O(nf(n))$ time, there is an algorithm to find at most $\lfloor \alpha^{-1} \rfloor$ points $p_1, ..., p_\ell$ such that each ball $B^C(p_i)$ of radius Cr around p_i has weight at least αw, and any ball of radius r around p_i with weight at least αw intersects some $B^C(p_i)$. Then, there is a $\frac{C+1}{1-\alpha} + 1 + \epsilon$-approximation to geometric 1-median in $O(\epsilon^{-1}nf(n)\log\log n + n\lceil \log n \rceil^{\lfloor \alpha^{-1} \rfloor})$ time.*

Proof. Suppose p is the (or a) geometric median for $a_1, ..., a_n$. Let r be the smallest radius of a ball centered at p that contains more than αn points, and let B be this ball. Then,

$$\sum_{i=1}^{n} \rho(p, a_i) \geq (1-\alpha)nr.$$

We show this means there is an $O_\epsilon(nf(n)\log\log\log n + n\lceil \log n \rceil^{\lfloor \alpha^{-1} \rfloor})$-time algorithm which returns a ball of radius between r and $(C + \epsilon(1-\alpha))r$ containing at least α of the points $a_1, ..., a_n$. To see why, first in $O(n \log n)$ time, we can determine a value s such that $r \leq s \leq 2\lceil \lg n \rceil r$, so $\frac{s}{2\lceil \lg n \rceil} \leq r \leq s$. For any $\frac{s}{2\lceil \lg n \rceil} \leq r' \leq s$, in $O(nf(n))$ time, according to our assumption, we can either show that there is no ball of radius r' containing at least αn points, or there is a ball of radius Cr' containing at least αn points.

Now, let $t = \frac{s}{2\lceil \lg n \rceil}$ and let $a = \lceil \frac{\log(2\lceil \lg n \rceil)}{\log(1+\epsilon(1-\alpha)/C)} \rceil$, so $a = O(\epsilon^{-1}\log\log n)$ as α, C are fixed. Then, if we consider the real numbers $t, t(1+\frac{\epsilon(1-\alpha)}{C}), t(1+\frac{\epsilon(1-\alpha)}{C})^2, ..., t(1+\frac{\epsilon(1-\alpha)}{C})^a$, we know that $t(1+\frac{\epsilon(1-\alpha)}{C})^{b-1} \leq r \leq t(1+\frac{\epsilon(1-\alpha)}{C})^b$ for some $1 \leq b \leq a$.

Suppose we knew this value of b. Then, we can find at most $\lfloor \alpha^{-1} \rfloor$ points $p_{b,1}, ..., p_{b,\ell}$ in $O(nf(n))$ time such that the ball of radius

$$Ct\left(1 + \frac{\epsilon(1-\alpha)}{C}\right)^b \leq Cr\left(1 + \frac{\epsilon(1-\alpha)}{C}\right) = Cr + \epsilon(1-\alpha)r$$

around some $p_{b,j}$ intersects B, i.e. $\rho(p, p_{b,j}) \leq (C+1)r + \epsilon(1-\alpha)r$ for some $p_{b,j}$.

Therefore, by the Triangle inequality,

$$\sum_{i=1}^{n} \rho(p_{b,j}, a_i) \leq n(C + 1 + \epsilon(1-\alpha))r + \sum_{i=1}^{n} \rho(p, a_i),$$

which means

$$\frac{\sum_{i=1}^{n}\rho(q,a_i)}{\sum_{i=1}^{n}\rho(p,a_i)} \leq 1+\frac{n(C+1+\epsilon(1-\alpha))r}{(1-\alpha)nr} = \frac{C+1}{1-\alpha}+1+\epsilon.$$

Therefore, we can first compute t in $O(n\lceil\log n\rceil^{\lfloor\alpha^{-1}\rfloor})$ time, and for each b run the algorithm to get some point p_b (or perhaps no such point) such that for some b and some $j \leq \lfloor\alpha^{-1}\rfloor$, $p_{b,j}$ is a $\frac{C+1}{1-\alpha}1+\epsilon$-approximation to geometric median. Since computing $\sum_i \rho(p_{b,j},a_i)$ takes $O(n)$ time and we need to find the smallest of these, overall we can find all possible $p_{b,j}$ in $O(\alpha^{-1}\cdot(\epsilon^{-1}\log\log n)nf(n))$ and compute the best $p_{b,j}$ in $O(\alpha^{-1}(\epsilon^{-1}\log\log n)n)$, for an overall runtime of

$$O\left(\epsilon^{-1}nf(n)\log\log n + n\lceil\log n\rceil^{\lfloor\alpha^{-1}\rfloor}\right)$$

where we are dropping the α^{-1} terms since α is fixed. ◀

Using the previous results, we can finally prove a strong lower bound on the General metric space 1-center clustering with outliers problem.

▶ **Theorem 13.** *For all fixed K,α, there does not exist a $((2K-3)(1-\alpha)-1)$ - approximation to the unweighted 1-center clustering with outliers problem in $O(n^{1+1/K})$ time.*

Proof. We first use Lemmas 10, 3, and 12. Set $\epsilon=\frac{1}{2}$ and $C=(2K-3)(1-\alpha)-1$. Suppose there is an algorithm for the unweighted problem for any arbitrary metric space with fraction α and approximation constant C that runs in $O(n^{1+1/K})$ time. Then, for any $\alpha'>\alpha$, there is an $O_{\alpha'}(n^{1+1/K})$-time algorithm that gets us to the conditions of Lemma 12 with fraction α' and constant $C+2$, by Lemmas 10 and 3. Then, there is a $\left(\frac{C+1}{1-\alpha'}+\frac{3}{2}\right)$-approximation to geometric 1-median in $O_{\alpha'}(n^{1+1/K}\log\log n + n\lceil\log n\rceil^{\lfloor\alpha^{-1}\rfloor})$ time, which is clearly an $O_{\alpha'}(n^{1+1/(K-0.5)})$-time algorithm. If we choose α' so that $\frac{C+1}{1-\alpha'}=\frac{C+1}{1-\alpha}+\frac{1}{2}$, then α' only depends on α, C, so there is a $\left(\frac{C+1}{1-\alpha}+2\right)$-approximation to geometric 1-median in $O(n^{1+1/(K-0.5)})$ time.

Now, we directly apply the main result of [7]. The main result of [7] states that for any fixed constant K', there is no $(2\lceil K'\rceil-1)$-approximation to geometric 1-median in $O(n^{1+1/K'})$ time - in fact, there is no such approximation even using $O(n^{1+1/K'})$ queries to distance, and we are assuming distance queries take $O(1)$ time. Then, letting $K'=K-\frac{1}{2}$, there is no $(2K-1)$-approximation to geometric median in $O(n^{1+1/(K-0.5)})$ time, and thus no $\left(\frac{C+1}{1-\alpha}+2\right)$-approximation to geometric 1-median in $O(n^{1+1/(K-0.5)})$ time. Therefore, there cannot be a C approximation to the unweighted 1-center clustering with outliers problem in time $O(n^{1+1/K})$. ◀

B Omitted Proofs

First, we prove Lemma 3 from Section 3.

Proof of Lemma 3. We induct on $\lfloor\alpha^{-1}\rfloor$. For $\alpha>\frac{1}{2}$ and some $\beta\geq\alpha$, we can in $O(ndf(n))$ time output p such that the ball of radius Cr around p contains at least βw total weight. But then since $\beta>\frac{1}{2}$, the second condition is true by default, so we are done. Also, if there is no ball of radius r containing αw weight, our algorithm may output some point, but in $O(nd)$ time we can verify and either output a ball of radius Cr containing αw weight, or output nothing.

Suppose $\alpha > \frac{1}{z+1}$ and we know it is true for all $\alpha' > \frac{1}{z}$. In $O_\alpha(ndf(n))$ time, we can find $B^C(p_1)$, a ball of radius Cr around some p_1 containing at least βw total weight for some $\beta \geq \alpha$. Again, if no such ball of radius r exists, we will either get nothing, in which case we end the program, or may happen to get a point p_1 such that $B^C(p_1)$ contains αw weight. Assuming we got a point in $O(nd)$ time we can remove all points in $B^C(p_1)$ by just checking all points' distances from p_1. Then, the remaining weight is $(1-\beta')w$ for some $\beta' \geq \beta$, and β' can be calculated in $O(nd)$ time.

If there exists a ball B of radius r that doesn't intersect $B^C(p_1)$, none of the points in B were removed, which means it has at least $\frac{\beta}{1-\beta'}$ of the remaining weight. Let $B^C(p_i)$ be the ball of radius Cr around p_i. We apply the induction hypothesis with fraction $\frac{\beta}{1-\beta'} > \frac{1}{z}$. It tells us in $O(ndf(n)(z-1))$ time we can find at most $z-1$ points $p_2, ..., p_\ell$ such that every ball of radius r containing at least αw weight either intersects $B^C(p_1)$ or it still contains at least $\frac{\beta}{1-\beta'}$ of the remaining weight, which means it intersects $B^C(p_i)$ for some $2 \leq i \leq \ell$.

If there does not exist a ball of radius r containing at least αw weight not intersecting $B^C(p_1)$, we will either output no points after p_1, or we may still output some points $p_2, ..., p_\ell$ such that $B^C(p_i)$ contains at least $\frac{\beta}{1-\beta'}$ of the remaining weight, or αw total weight. But since every ball of radius r containing at least αw weight intersects $B^C(p_1)$, we are done. ◀

Next, we prove the base cases of $\lfloor \alpha^{-1} \rfloor = 1$ and $C = 1$ for Theorem 8.

▶ **Theorem 14.** *For $\alpha > \frac{1}{2}$, suppose we are trying to solve the weighted 1-center clustering problem in a general metric space, but now assuming r is unknown. Then, for any positive integer C, we can find a point p such that the ball of radius $2Cr$ around p contains at least αw of the weight in $O(Cn^{1+1/C})$ time, assuming $n = m^C$ for some integer m. As an obvious consequence, every ball of radius r containing at least αw of the weight must intersect the ball of radius $2Cr$ around p.*

Proof. For $C = 1$, we compute for each a_i the quantities $\rho(a_i, a_1), ..., \rho(a_i, a_n)$ and let r_i be the smallest real number such that the ball of radius r_i around a_i contains at least αw total weight. This can be computed for each i in $O(n)$ time using standard algorithms for weighted median, and thus takes a total of $O(n^2)$ time for all i. Then, if some r_i equals $\min(r_1, ..., r_n)$, the ball of radius r_i around a_i contains at least αw total weight, and $r_i \leq 2r$ since otherwise, there is a ball of radius r around some p in the metric space containing at least αw total weight, which means the ball of radius $2r$ around around some p_j in that radius r ball must contain at least αw total weight, so $r_i \leq 2r$. This proves our claim for $C = 1$.

Assume there is an algorithm that works for $C-1$. Then, split the n points into m blocks $D_1, ..., D_m$ of size m^{C-1}. For each block D_i, we can return $p_i \in a_{D_i}$ such that if there is a ball of radius r that when intersected with a_{D_i} contains at least αw_{D_i} weight, then the ball of radius $2(C-1)r$ around p_i intersected with a_{D_i} contains at least αw_{D_i} weight. Moreover, we can determine $p_1, ..., p_m$ in $O((C-1)(n/m)^{1+1/(C-1)} \cdot m) = O((C-1)n^{1+1/C})$ time.

If B is a ball of radius r containing at least α of the total weight, then there exists some $1 \leq k \leq m$ such that $w_{D_k} > 0$ and the total weight of $a_{D_k} \cap B$ is at least αw_{D_k}. Since the ball of radius $(2C-2)r$ around p_k contains at least αw_{D_k} weight when intersected with a_{D_k}, and since $\alpha > \frac{1}{2}$, the ball of radius $(2C-2)r$ around p_k must intersect B. Therefore, the ball of radius $2Cr$ around p_k contains B and thus contains at least αw weight when intersected with $a_{[n]}$.

This means after we get our points $p_1, ..., p_m$, the ball of radius $2Cr$ around at least one of the p_i's must have at least αw total weight. We determine $r_1, ..., r_m$ where r_i is the radius of the smallest ball around p_i containing at least αw of the original weight, which can be done in $O(n)$ time for each i since weighted median can be solved in $O(n)$ time. Doing

this for each p_i takes $O(nm) = O(n^{1+1/C})$ time, and if $r_i = min(r_1, ..., r_m)$ for some i, then clearly $r_i \leq 2Cr$. Therefore, this takes $O((C-1)n^{1+1/C}) + O(n^{1+1/C}) = O(Cn^{1+1/C})$ time total, so our induction step is complete. ◀

▶ **Lemma 15.** *For any $\alpha > 0$, say we are trying to solve weighted 1-center clustering with outliers in a general metric space, with r unknown. In $O(\alpha^{-1}n^2)$ time we can find $\ell \leq \lfloor \alpha^{-1} \rfloor$ points $p_1, ..., p_\ell$ with corresponding radii $s_1, ..., s_\ell$ such that the ball of radius s_i around p_i contains at least αw weight. Moreover, any ball of radius r containing at least αw weight will intersect at least one ball of radius s_i around p_i where $s_i \leq 2r$.*

Proof. Define $y = \alpha w$. Like in Theorem 14, we find for each $a_1, ..., a_n$ values $r_1, ..., r_n$ such that r_i is the smallest radius around a_i of a ball containing at least αw total weight, and these can all be done in $O(n^2)$ time. Let p_1 be the point a_i with smallest corresponding r_i, and let s_1 be the corresponding r_i. Clearly, $r_i \leq 2r$ and the total weight of the points in the ball of radius r_i around p_1 is at least y. Remove all the points in this ball. Repeat this procedure (for the same y, not α times the new total weight) until we have $p_1, ..., p_\ell$ and the remaining points have weight less than y. This procedure clearly takes $O(\alpha^{-1}n^2)$ time.

Suppose some ball B contains at least αw weight but does not intersect a ball of radius s_i around r_i for any i such that $s_i \leq 2r$. Then, suppose j is the largest integer such that $s_i \leq 2r$ for all $i \leq j$. Either $j = \ell$ or $s_{j+1} > 2r$. If $j = \ell$, then the remaining points have weight less than y, which makes no sense since B has weight at least y and does not intersect any of the balls we created. If $s_{j+1} > 2r$, we would have picked a different ball. This is because if $a_k \in B$, the ball of radius $2r$ around a_k contains at least αw weight, so we would have picked a_k as our point p_{j+1} instead. Thus, we are done. ◀

Robust Online Speed Scaling With Deadline Uncertainty

Goonwanth Reddy
Department of Electrical Engineering,
Indian Institute of Technology, Madras, Chennai, India
ngoonwanth@gmail.com

Rahul Vaze
School of Technology and Computer Science,
Tata Institute of Fundamental Research, Mumbai, India
vaze@tcs.tifr.res.in

Abstract

A speed scaling problem is considered, where time is divided into slots, and jobs with payoff v arrive at the beginning of the slot with associated deadlines d. Each job takes one slot to be processed, and multiple jobs can be processed by the server in each slot with energy cost $g(k)$ for processing k jobs in one slot. The payoff is accrued by the algorithm only if the job is processed by its deadline. We consider a robust version of this speed scaling problem, where a job on its arrival reveals its payoff v, however, the deadline is hidden to the online algorithm, which could potentially be chosen adversarially and known to the optimal offline algorithm. The objective is to derive a robust (to deadlines) and optimal online algorithm that achieves the best competitive ratio. We propose an algorithm (called min-LCR) and show that it is an optimal online algorithm for any convex energy cost function $g(.)$. We do so without actually evaluating the optimal competitive ratio, and give a general proof that works for any convex g, which is rather novel. For the popular choice of energy cost function $g(k) = k^\alpha, \alpha \geq 2$, we give concrete bounds on the competitive ratio of the algorithm, which ranges between 2.618 and 3 depending on the value of α. The best known online algorithm for the same problem, but where deadlines are revealed to the online algorithm has competitive ratio of 2 and a lower bound of $\sqrt{2}$. Thus, importantly, lack of deadline knowledge does not make the problem degenerate, and the effect of deadline information on the optimal competitive ratio is limited.

2012 ACM Subject Classification Theory of computation → Scheduling algorithms

Keywords and phrases Online Algorithms, Speed Scaling, Greedy Algorithms, Scheduling

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.22

1 Introduction

Energy efficient transmission of packets in communication systems or processing of jobs in microprocessors (and similar applications) is a fundamental resource allocation problem. A job/packet can be processed/transmitted by a server 'fast' but only at the cost of higher energy consumption. Typically, the energy cost is a convex function of the server speed, (e.g., a popular choice is x^α for $\alpha \geq 2$) and the general objective is two fold: maximize the profit from processing jobs while incurring minimum energy cost. The profit can either be the payoff accrued on processing a job or some function of the inverse of the processing time. This problem is known as *speed scaling* problem in literature, where the speed of the server is the tunable parameter which determines the profit and the energy consumption.

Speed scaling problem has been considered with both the infinite speed model as well as bounded speed model, where in the former case, the server is allowed to scale its speed

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018).
Editors: Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer; Article No. 22; pp. 22:1–22:17

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

anywhere in $[0, \infty)$, while in the latter case, it is bounded by a fixed constant. Infinite speed model allows processing of all jobs, and the main concern is to minimize energy consumption, while in the bounded speed model, both maximizing the profit and minimizing the energy consumption is a challenge.

Another broad classification considered with the speed scaling problem is with respect to deadlines. In the case when jobs do not have deadlines, a typical objective is to minimize a linear combination of the sum of the flow time (completion time minus the arrival time) and the energy consumption for each job. When jobs also specify a deadline, under the bounded speed model, the objective is to process jobs that maximize the profit (accrued only from jobs that are processed by their deadline) while minimizing the energy cost.

An alternate speed scaling model (which we consider in this paper) is a discretized one, where time is divided into slots, and set of jobs S with payoff $v_i, i \in S$ arrive at the beginning of the slot with associated deadlines $d_i, i \in S$. Each job takes one slot to be processed, and multiple jobs can be processed by the server in each slot with energy cost $g(k)$ for processing k jobs in one slot. The payoff is accrued by the algorithm only if the job is processed by its deadline, and the objective is to maximize the sum of the profit (payoff minus energy cost) of the processed jobs.

One limiting aspect of almost all literature on speed scaling with job deadlines is the need for the exact knowledge of deadlines. The derived results critically depend on exact deadline information, and are not robust to even a small uncertainty. In modern applications, the job deadlines could be time varying, could potentially depend on other jobs or their completion times, or may not be precisely known on the job arrival. Towards addressing this critical aspect as well as to generalize the model, we consider a robust version of the discretized speed scaling problem, where a job on its arrival reveals its payoff v, however, the deadline is hidden to the online algorithm, which could potentially be chosen adversarially and known to offline optimal algorithm. This approach will lead to the derivation of robust online algorithms for speed scaling and provides a means to quantify the fundamental effect of deadline knowledge on speed scaling. We note that the robust approach is useful only if the problem itself does not become degenerate, in the sense that no online algorithm can achieve a reasonable competitive ratio.

For the considered robust model, we propose a simple online algorithm, called the minimum-local competitive ratio (min-LCR) algorithm, that does not need any deadline information. Let $c_k = g(k) - g(k-1)$ be the effective energy cost for processing the k^{th} job. Algorithm min-LCR indexes all the available jobs at each slot in the non-increasing order of their payoffs v, and computes the profit p_ℓ it will make if it processes $1 \leq \ell \leq m$ jobs, where m is such that $v_m - c_m > 0$ and $v_{m+1} - c_{m+1} < 0$. It also presumes a worst case scenario for the deadlines (since it does not know the exact deadlines) where the ℓ-chosen jobs for processing have deadlines infinity while the left-over jobs (other than the ℓ chosen jobs) have deadlines that expire in the current slot. Let o_ℓ be the profit that can be made by an offline algorithm under the knowledge of the worst choice of deadlines for the current set of available jobs, assuming no further jobs arrive. We define $\mathsf{LCR}_\ell = \frac{o_\ell}{p_\ell}$ for $1 \leq \ell \leq m$, which has the interpretation of local competitive ratio (ratio of the optimal offline and the simple online algorithm under the worst case deadlines given no further jobs arrive), and the algorithm chooses to process ℓ jobs for which LCR_ℓ is minimum. Note that always choosing $\ell = m$ will correspond to a natural greedy algorithm for this problem.

1.1 Contributions

We make the following contributions in this paper.

- We show that the proposed Algorithm min-LCR is an optimal online algorithm for any convex energy cost function $g(.)$. We do so without actually finding the optimal competitive ratio, which is fundamentally different than the typical proof strategy used in the analysis of online algorithms. To the best of our knowledge, such a result is not available in the speed scaling literature to achieve the optimal competitive ratio even when deadlines are exactly known.
- The optimality of the min-LCR algorithm is shown via first constructing a lower bound which with deadline uncertainty is easier compared to prior work when deadlines are revealed, since deadlines can be chosen arbitrarily that are not revealed to any online algorithm. The more challenging task is to show that the proposed min-LCR algorithm actually achieves the lower bound even when it works without any knowledge about the deadlines of any of the jobs.
- To ensure that the considered robust speed scaling problem is not degenerate in the sense that the adversarial deadlines choices that are unknown to the online algorithm make the competitive ratio of any online algorithm arbitrarily large, we provide concrete analysis for the most popular energy cost function of $g(k) = k^\alpha$. The min-LCR algorithm involves finding the optimal number of jobs to process among the available ones that minimizes the LCR which can be computationally extensive. To simplify the complexity, we also consider a simplified version of the min-LCR algorithm, where exactly $k = \lfloor \beta m \rfloor$ or $k = \lceil \beta m \rceil$ jobs are processed in each slot depending on whichever one has lower LCR, where β is the solution of the equation $x^\alpha + x^{\alpha-1} - 1 = 0$. We also consider the natural greedy special case of min-LCR that always processes m jobs, where m has been defined in the min-LCR algorithm description as above.
- We show that the optimal competitive ratio for energy cost function of $g(k) = k^\alpha$ with $\alpha = 2$ is $\phi + 1$ ($\phi = 1/\delta, \delta = \frac{\sqrt{5}-1}{2}$), and is achieved by the simplified min-LCR algorithm. In comparison, when deadlines are known to the online algorithm, for $\alpha \geq 2$ the best known online algorithm has competitive ratio of 2 and the best known lower bound is $\sqrt{2}$.
- For $\alpha \geq 2.5$, the competitive ratio of the simplified min-LCR algorithm is at most $\phi + 1$, and for any $\alpha \geq 2$, the competitive ratio of the greedy algorithm (min-LCR algorithm with $\ell = m$ always) is at most 3. We also derive a lower bound on the competitive ratio of $\sqrt{2} + 1$ for all $\alpha \geq 2$.
- Thus, we show that for the energy cost function of $g(k) = k^\alpha$, lack of deadline knowledge reduces the optimal competitive ratio by a factor of at most $3/\sqrt{2}$. Thus, the loss in performance because of deadline uncertainty is limited, and precludes the possibility that the considered robust model is inherently weak, and the power of any online algorithm is seriously limited. Moreover, it also obviates the need to study the unknown but stochastic deadline case, since the loss in competitive ratio from fully known to completely adverarial is very small.

1.2 Related Work

Starting with [19], there has been a long line of work on optimal speed scaling for servers with unbounded speed. For energy cost $g(k) = k^\alpha, \alpha > 1$, a $2^{\alpha-1}\alpha^\alpha$-competitive algorithm was derived in [19], whose competitive ratio was subsequently improved upon in [6] to $2\left(\frac{\alpha}{\alpha-1}\right)^\alpha \exp(1)^\alpha$, and eventually in [5] to $4^\alpha/(2\sqrt{\exp(1)\alpha})$.

For the bounded speed case, where all jobs cannot be processed, [10] first derived an online algorithm that is 14-competitive algorithm for throughput (number of processed jobs) and $\alpha^\alpha + \alpha^2 4^\alpha$-competitive for energy. Subsequently, the throughput competitiveness was improved to 4 in [4], which is also the best possible [9].

In addition to speed scaling, an additional feature of *sleeping* was introduced in [15], where all jobs can be processed with $(2^{2\alpha-2}\alpha^\alpha + 2^{\alpha-1} + 2)$-competitiveness in terms of energy, which was improved upon in [14] to get $\alpha^\alpha + 2$-competitiveness in terms of energy under the infinite speed model when all jobs can be processed, and 4-competitive algorithm for throughput (number of processed jobs) and $(\alpha^\alpha + \alpha^2 4^\alpha + 2)$-competitive for energy in the bounded server model.

The no-deadline model where the objective function is to minimize the flow time plus the energy has been considered widely [16, 18, 7], with the most general result obtained in [7] that gives a constant competitive algorithm for all energy cost functions. The speed scaling problem in the no-deadline model, where some information about jobs (either value/weight/density) is hidden is called the non-clairvoyant setting and has been addressed in literature starting with [11] and followed up in [2]. Speed scaling with multiple processors has also been considered in [1] and with non-clairvoyant setting in [13], while modern applications for speed scaling are being addressed in [8], where energy is derived from solar cells for renewable energy harvesting.

The discrete model studied in this paper was first considered in [12], where jobs deadlines are known to the online algorithm, which proposed an online algorithm that is 2-competitive, using the ideas from online request matching [17]. An associated lower bound (easy to construct) on the competitive ratio for this problem is $\sqrt{2}$.

As far as we know the speed scaling problem when jobs have hard deadlines that are not exactly known to the online algorithm has not been considered in literature. In load balancing literature, unknown job deadline case is referred to as scheduling for temporary tasks, where the duration for which a job lasts is unknown [3]. In load-balancing, however, each job has to be scheduled as soon as it arrives, and the only decision variables are : which server to be assigned for each job and the dynamic server speed.

2 Problem Definition

We consider a discrete time system, where time is divided in discrete slots. A sequence $\boldsymbol{\sigma} = J_1, \ldots, J_n$ of jobs arrives causally, where job J_i arrives at slot/time a_i and must be finished by a deadline of d_i slots starting from a_i or is dropped. We assume $a_i \leq a_{i+1}$ for $1 \leq i < n$. Each job takes one slot to be processed, and if processed before its deadline, job J_i accumulates a payoff/value v_i. A job is *available* at slot t if its absolute deadline is after slot $t-1$, and is *expired* otherwise excluding the already processed jobs. The server can work at variable speed, and can process $k \geq 0$ jobs in any slot by incurring a cost of $g(k)$, where $g(.)$ is a convex function, e.g., $g(k) = k^\alpha, \alpha > 1$.

In a significant departure from prior work on speed scaling, we consider the robust setting, where the online algorithm does not know the deadline for any job, which could potentially be chosen by an adversary. This allows us to model the deadline uncertainty, derive a robust online algorithm, and provide a means to quantify the effect of deadline knowledge on speed scaling. Thus, the information that any online algorithm has at slot t is the set of jobs that have not expired by then, and their respective values. We, however, let the offline optimal algorithm to know the exact deadline and the respective payoff non-causally, to consider the worst case model.

We consider only the deterministic algorithm setting, where on a sequence of jobs $\boldsymbol{\sigma}$, let the set of jobs processed at slot j by an algorithm ALG be P_j. Then the overall profit for ALG is

$$C_{\mathsf{ALG}}(\boldsymbol{\sigma}) = \sum_{j=1}^{\mathsf{last}} (v_{P_j} - g(|P_j|)),$$

where $v_{P_j} = \sum_{i \in P_j} v_j$, and last is the last slot at which Algorithm ALG processes any job. Let $\mathsf{r}_{\mathsf{ALG}}(\boldsymbol{\sigma}) = \frac{C_{\mathsf{OFF}}(\boldsymbol{\sigma})}{C_{\mathsf{ALG}}(\boldsymbol{\sigma})}$, where OFF is the offline optimal algorithm that is allowed to know the sequence $\boldsymbol{\sigma}$ including the job deadlines non-causally. The objective is to find the optimal competitive ratio

$$\mathsf{r}^\star = \min_{\mathsf{ALG}} \max_{\boldsymbol{\sigma}} \mathsf{r}_{\mathsf{ALG}}(\boldsymbol{\sigma}), \tag{1}$$

and the optimal online algorithm achieving $\mathsf{r}^\star$ be OPT. To reiterate, $\min_{\mathsf{ALG}}$ is over all deterministic online algorithms that do not have deadline information of jobs, and make decisions causally at each slot depending on the values of the available jobs only.

▶ **Definition 1.** The effective/incremental cost of processing the k^{th} job in any slot is $c_k = g(k) - g(k-1)$. Since $g(k)$ is convex, $c_k > 0 \ \forall \ k$.

▶ **Definition 2.** For two inputs $\boldsymbol{\sigma}_1$ and $\boldsymbol{\sigma}_2$, $\boldsymbol{\sigma}_1 \cup \boldsymbol{\sigma}_2$ corresponds to input where for each slot t, the set of jobs is the union of job arriving at slot t in $\boldsymbol{\sigma}_1$ and $\boldsymbol{\sigma}_2$.

▶ **Lemma 3.** *For any convex function $g(.)$, for any two inputs $\boldsymbol{\sigma}_1$ and $\boldsymbol{\sigma}_2$, $C_{\mathsf{OFF}}(\boldsymbol{\sigma}_1 \cup \boldsymbol{\sigma}_2) \leq C_{\mathsf{OFF}}(\boldsymbol{\sigma}_1) + C_{\mathsf{OFF}}(\boldsymbol{\sigma}_2)$.*

Proof. Let the profit (payoff minus the energy cost) accrued in $C_{\mathsf{OFF}}(\boldsymbol{\sigma}_1 \cup \boldsymbol{\sigma}_2)$ corresponding to the processing of jobs belonging to $\boldsymbol{\sigma}_i$ be p_i. From convexity of the cost function $g(.)$, we have $p_i \leq C_{\mathsf{OFF}}(\boldsymbol{\sigma}_i)$ and the result follows since $C_{\mathsf{OFF}}(\boldsymbol{\sigma}_1 \cup \boldsymbol{\sigma}_2) = p_1 + p_2$. ◀

3 min-LCR algorithm

▶ Remark. Since $c_k > 0$ and increasing in k, m (the largest number of jobs that can be processed, where each job has a positive profit) is well defined in min-LCR algorithm.

▶ Remark. A natural Greedy algorithm is a special case of the min-LCR algorithm when $i^\star(\tau) = m$ at all slots τ.

The basic idea behind the min-LCR algorithm has been described in the introduction. To be specific, the online algorithm's profit is $P_{i,\tau}$ if it processes i jobs in slot τ without the knowledge of the deadlines of the available jobs. The online algorithm presumes that possibly no further jobs are going to arrive, and the i-chosen jobs for processing by the algorithm have deadlines as infinity, while the left-over jobs (other than the i chosen jobs) have deadlines that expire in the current slot. The profit of the OFF under this presumption is $o_{i,\tau} = M_{i,\tau} + C_{\mathsf{Greedy}}(i,\tau)$, where $M_{i,\tau}$ is the profit OFF can accrue by processing i highest valued jobs one in each slot starting from slot $\tau + 1$ if their deadlines are infinity, while $C_{\mathsf{Greedy}}(i,\tau)$ is the largest profit possible by processing the set of jobs other than the i highest valued jobs in slot τ itself. The algorithm chooses i that minimizes the ratio of $\frac{o_{i,\tau}}{P_{i,\tau}}$ which essentially is the local competitive ratio at slot τ.

▶ **Theorem 4.** *Algorithm min-LCR is an optimal online algorithm that achieves the optimal competitive ratio* (1).

Algorithm 1: min-LCR algorithm.

Input : Sequence $\boldsymbol{\sigma} = J_1, \ldots, J_n$
At slot τ, consider the union of all the non-processed jobs by the algorithm so far that are available and the newly arriving jobs in slot τ, and call it $E(\tau)$
Arrange the jobs in $E(\tau)$ in non-increasing order of their payoffs/value v
Let $E(\tau)^i = \{\text{first } \ i \text{ jobs in } E(\tau)\}$ and $E^{i-}(\tau) = E(\tau) \backslash E(\tau)^i$
Let $v^{(i)}$ be the value of job of $E(\tau)$ with the i^{th} highest value and $V^{(i)}$ be the sum of the values of the first i jobs with highest values
Let $m = \max_{\{v^{(j)} - [g(j) - g(j-1)] > 0\}} j$
for $i = 1 : m$ **do**
  Let
  $M_{i,\tau} = V^{(i)} - ig(1), P_{i,\tau} = V^{(i)} - g(i), C_{\mathsf{Greedy}}(i,\tau) = \max_{j \in E^{i-}(\tau)} \{V^{(j)} - g(j)\}$
  $\mathsf{LCR}_i(\tau) = \dfrac{M_{i,\tau} + C_{\mathsf{Greedy}}(i,\tau)}{P_{i,\tau}}$
end
Let $i^\star(\tau) = \arg\min_{i=1,\ldots,m} \mathsf{LCR}_i(\tau)$
Process first $i^\star(\tau)$ jobs of $E(\tau)$ at slot τ, call the processed set of jobs $L^p_{\boldsymbol{\sigma}}(\tau)$.

To prove Theorem 4, we show that the competitive ratio of Algorithm min-LCR on input $\boldsymbol{\sigma}$ is at most $\max_\tau \mathsf{LCR}_{i^\star(\tau)}(\tau)$ in Lemma 5, and $\max_\tau \mathsf{LCR}_{i^\star(\tau)}(\tau) \leq \mathsf{r}^\star$ in Lemma 6.

▶ **Lemma 5.** *For any input $\boldsymbol{\sigma}$, $\mathsf{r}_{min-LCR}(\boldsymbol{\sigma}) \leq \max_\tau \mathsf{LCR}_{i^\star(\tau)}(\tau)$.*

Proof. Let $L^p_{\boldsymbol{\sigma}}(\tau)$ be the jobs processed by Algorithm min-LCR on input $\boldsymbol{\sigma}$ at slot τ, and $\bigcup_{\tau=1}^{\mathsf{last}} L^p(\tau) = L^p$, where last is the last slot at which Algorithm min-LCR processes any job.

$$
\begin{aligned}
C_{\mathsf{OFF}}(\boldsymbol{\sigma}) &= C_{\mathsf{OFF}}(L^p \cup \boldsymbol{\sigma} \backslash L^p), \\
&\overset{(a)}{\leq} C_{\mathsf{OFF}}(L^p) + C_{\mathsf{OFF}}(\boldsymbol{\sigma} \backslash L^p) = C_{\mathsf{OFF}}\left(\bigcup_{\tau=1}^{\mathsf{last}} L^p(\tau) \right) + C_{\mathsf{OFF}}(\boldsymbol{\sigma} \backslash L^p), \\
&\overset{(b)}{\leq} \sum_{\tau=1}^{\mathsf{last}} C_{\mathsf{OFF}}(L^p(\tau)) \ + \ C_{\mathsf{OFF}}(\boldsymbol{\sigma} \backslash L^p), \\
&= \sum_{\tau=1}^{\mathsf{last}} C_{\mathsf{OFF}}(L^p(\tau)) \ + \ \sum_{\tau=1}^{\mathsf{last}} C_{\mathsf{OFF}}(\boldsymbol{\sigma} \backslash L^p)_\tau, \qquad (2)
\end{aligned}
$$

where (a) and (b) follow from sub-additivity Lemma (Lemma 3), and where $C_{\mathsf{OFF}}(\boldsymbol{\sigma} \backslash L^p)_\tau$ is the profit obtained by the OFF algorithm in slot τ when the input is only $\boldsymbol{\sigma} \backslash L^p$. Note that this is not true for concave cost functions since sub-additivity lemma (Lemma 3) does not hold. Equation 2 follows from the fact that min-LCR terminates at time slot last leaving behind only unprofitable jobs for slots after last. Hence it is not profitable to process any unprocessed job out of set $\boldsymbol{\sigma} \backslash L^p$ after last.

In order to upper bound $C_{\mathsf{OFF}}(\boldsymbol{\sigma})$ we first observe that the maximum profit that can be made from jobs in set $L^p(\tau)$ is by processing them alone in distinct slots with energy cost of $g(1)$, hence

$$
C_{\mathsf{OFF}}(L^p(\tau)) \leq \sum_{i \in L^p(\tau)} (v_i - g(1)). \qquad (3)
$$

Since $L^p(\tau)$ is the set consisting of first $i^\star(\tau)$ jobs of $E(\tau)$, and $|L^p(\tau)| = i^\star(\tau)$, we get $\sum_{i\in L^p(\tau)} v_i = V^{i^\star(\tau)}$ and $\sum_{i\in L^p(\tau)} g(1) = i^\star(\tau)g(1)$. Hence from (3), we get

$$C_{\mathsf{OFF}}(L^p(\tau)) \le V^{i^\star(\tau)} - i^\star(\tau)g(1) = M_{i^\star,\tau}, \tag{4}$$

where the last inequality follows from the definition of $M_{i^\star,\tau}$ from Algorithm min-LCR. To bound $C_{\mathsf{OFF}}(\boldsymbol{\sigma}\backslash L^p)_\tau$ in (2), we observe that any job among $\boldsymbol{\sigma}\backslash L^p$ that OFF can process at time τ was also available to min-LCR($\boldsymbol{\sigma}$) at time τ but was not processed by min-LCR. Since the maximum profit that can be accrued from all the available packets to min-LCR at time τ is $C_{Greedy}(i^\star,\tau)$ the profit made by OFF($\boldsymbol{\sigma}\backslash L^p$) at time τ must be less than this value.

The set $\boldsymbol{\sigma}\backslash L^p$ consists of all jobs of $\boldsymbol{\sigma}$ that are not processed by the min-LCR algorithm at any slot during its operation. $(\boldsymbol{\sigma}\backslash L^p)_\tau$ is a set of elements of $\boldsymbol{\sigma}\backslash L^p$ which arrived at or before slot τ and whose deadline is after slot $\tau-1$. In comparison, the set $E(\tau)\backslash L^p(\tau)$ is the union of jobs available at slot τ that are never processed by the min-LCR algorithm and the available jobs at slot τ that are processed by the min-LCR algorithm in some slot after slot τ.

Thus,

$$(\boldsymbol{\sigma}\backslash L^p)_\tau \subseteq E(\tau)\backslash L^p(\tau), \tag{5}$$

Similar to the definition of $C_{\mathsf{OFF}}(\boldsymbol{\sigma}\backslash L^p)_\tau$ in (2), let $C_A(\boldsymbol{\sigma}\backslash L^p)_\tau$ be the profit obtained by the algorithm A in slot τ when the input is only $\boldsymbol{\sigma}\backslash L^p$. Then, $C_{\mathsf{OFF}}(\boldsymbol{\sigma}\backslash L^p)_\tau \le \max_{A\in\mathsf{OFF-ALG}} C_A(\boldsymbol{\sigma}\backslash L^p)_\tau$, where the maximization is over all the offline algorithms. Hence, using (5), we get

$$C_{\mathsf{OFF}}(\boldsymbol{\sigma}\backslash L^p)_\tau \le \max_{A\in\mathsf{OFF-ALG}} C_A(E(\tau)\backslash L^p(\tau))_\tau = C_{\mathsf{Greedy}}(i^\star,\tau), \tag{6}$$

where the last equality is derived as follows. Let $A^\star = \arg\max_{A\in\mathsf{OFF-ALG}} C_A(E(\tau)\backslash L^p(\tau))_\tau$ and say it processes some k jobs with values $v_1,\ldots,v_k$ belonging to set $E(\tau)\backslash L^p(\tau)$ in slot τ, then

$$\begin{aligned}\max_{A\in\mathsf{OFF-ALG}} C_A(E(\tau)\backslash L^p(\tau))_\tau &= \sum_{i=1}^{k} v_i \ - g(k),\\ &\overset{(a)}{\le} \sum_{i=1}^{k} v^{(i)} \ - g(k) \ = V^{(k)} - g(k),\\ &\le \max_k\{V^{(k)} - g(k)\} \overset{(b)}{=} C_{\mathrm{Greedy}}(i^\star,\tau),\end{aligned}$$

where (a) follows since sum of the values of any set of k jobs is less than those of the k highest valued jobs (where $v^{(i)}$ is the value of the i^{th} highest valued job), and (b) follows from the definition of $C_{\mathrm{Greedy}}(i^\star,\tau)$ in Algorithm min-LCR since the jobs chosen by $A^\star$ belong to the set $E(\tau)\backslash L^p(\tau)$.

Hence, using (4) and (6), we have from (2),

$$C_{\mathsf{OFF}}(\boldsymbol{\sigma}) \le \sum_{\tau=1}^{\mathsf{last}} M_{i^\star(\tau),\tau} + \sum_{\tau=1}^{\mathsf{last}} C_{\mathsf{Greedy}}(i^\star(\tau),\tau).$$

From the definition of the Algorithm min-LCR, the profit made by it at slot τ is $P_{i^\star(\tau),\tau} = V^{(i^\star(\tau))} - g(i^\star(\tau))$ by processing $i^\star(\tau)$ jobs at slot τ. Thus, the competitive ratio

of min-LCR on input ($\boldsymbol{\sigma}$) algorithm is at most

$$\mathsf{r}_{LCR}(\boldsymbol{\sigma}) = \frac{C_{\mathsf{OFF}}(\boldsymbol{\sigma})}{C_{\mathrm{LCR}}(\boldsymbol{\sigma})} \leq \frac{\sum_{\tau=1}^{\mathsf{last}}(M_{i^\star(\tau),\tau} + C_{\mathsf{Greedy}}(i^\star(\tau),\tau))}{\sum_{\tau=1}^{\mathsf{last}} P_{i^\star(\tau),\tau}} \leq \max_{\tau} \frac{M_{i^\star(\tau),\tau} + C_{\mathsf{Greedy}}(i^\star(\tau),\tau)}{P_{i^\star(\tau),\tau}},$$
$$= \max_{\tau} \mathsf{LCR}_{i^\star(\tau)}(\tau). \tag{7}$$

◀

3.1 Lower Bound

Next, to complete the proof of Theorem 4, we show that the optimal competitive ratio $\mathsf{r}^\star$ is lower bounded by $\max_\tau \mathsf{LCR}_{i^\star(\tau)}(\tau)$ for any input $\boldsymbol{\sigma}$ and any slot τ. The main idea in the proof is that if there exists an optimal online algorithm OPT which has a strictly better competitive ratio ($\mathsf{r}^\star$) than min-LCR, then there must exist an input ($\boldsymbol{\sigma}_1$) at which $\frac{\mathsf{OFF}(\boldsymbol{\sigma}_1)}{LCR(\boldsymbol{\sigma}_1)} > \mathsf{r}^\star$ and $\frac{\mathsf{OFF}(\boldsymbol{\sigma}_1)}{\mathsf{OPT}(\boldsymbol{\sigma}_1)} \leq \mathsf{r}^\star$. Using this input we then construct another input ($\boldsymbol{\sigma}_2$) for which $\frac{\mathsf{OFF}(\boldsymbol{\sigma}_2)}{\mathsf{OPT}(\boldsymbol{\sigma}_2)} > \mathsf{r}^\star$ contradicting the assumption that competitive ratio of OPT is $\mathsf{r}^\star$.

▶ **Lemma 6.** *$\mathsf{r}^\star \geq \max_\tau \mathsf{LCR}_{i^\star(\tau)}(\tau)$ for any input $\boldsymbol{\sigma}$ and time slot τ.*

Contradiction. Let the hypothesis H_1 : $\exists\ \tau, \boldsymbol{\sigma}_1$ such that $\mathsf{LCR}_{i^\star(\tau)}(\tau) > \mathsf{r}^\star$. For ease of exposition in this proof, let $\mathsf{LCR}^{(\tau)}_{\boldsymbol{\sigma}_1} = \mathsf{LCR}_{i^\star(\tau)}(\tau)$. Consider the set of jobs $E(\tau)$ at slot τ that is the union of all the non-processed and non-expired jobs till slot $\tau - 1$ by the Algorithm min-LCR and the newly arrived jobs at slot τ. Let $v_1, \ldots, v_{|E(\tau)|}$ be the values of these jobs, where job i arrived at slot a_i and has deadline d_i.

We construct another input sequence $\boldsymbol{\sigma}_2$ that consists of $|E(\tau)|$ jobs with values $v_1, \ldots, v_{|E(\tau)|}$. All the jobs of $\boldsymbol{\sigma}_2$ arrive at slot 1, and the deadlines for each job is equal to the $d_i - \tau - a_i$. Important to note that d_i is allowed to be arbitrary and unknown to the online algorithm while known to the OFF in both $\boldsymbol{\sigma}_1$ and $\boldsymbol{\sigma}_2$.

Consider a new hypothesis H_2 : Optimum online algorithm OPT on input $\boldsymbol{\sigma}_2$ has competitive ratio lower than $\mathsf{LCR}^{(1)}_{\boldsymbol{\sigma}_2}$. It is easy to check that from the min-LCR algorithm definition, that $\mathsf{LCR}^{(1)}_{\boldsymbol{\sigma}_2} = \mathsf{LCR}^{(\tau)}_{\boldsymbol{\sigma}_1}$. From hypothesis H_1, $\mathsf{LCR}^{(\tau)}_{\boldsymbol{\sigma}_1} > \mathsf{r}^\star$, which implies that there exists an optimum online algorithm OPT that on input $\boldsymbol{\sigma}_2$ has competitive ratio lower than $\mathsf{LCR}^{(1)}_{\boldsymbol{\sigma}_2}$, since $\mathsf{r}^\star$ is achievable. Hence $H_1 \implies H_2$. Now we will contradict hypothesis H_2.

Let the optimal online algorithm OPT on input $\boldsymbol{\sigma}_2$ process any $1 \leq k \leq |E(\tau)|$ jobs at slot 1, leaving the remaining jobs for later slots. Since the deadlines d_i are arbitrary, let the true deadlines for the k jobs (whatever the choice of k may be for OPT) that were processed by OPT in slot 1 to be ∞, while keeping the deadlines for all jobs other than k selected ones to be slot 1 itself. Thus, there are no available jobs at slot 2 for OPT. Thus, the OPT can make the maximum profit by sending the k highest valued jobs of $\boldsymbol{\sigma}_2$ or $E(\tau)$ in slot 1.

Given that offline optimal OFF knows the deadlines, OFF processes the k jobs chosen by OPT for processing in slot 1 in k slots starting from the second slot individually, and among the rest of $|E(\tau)| - k$ jobs processes as many jobs in slot 1 to maximize its profit in slot 1, which by definition is $C_{\mathsf{Greedy}}(k, 1)$. Thus, $C_{\mathsf{OFF}}(\boldsymbol{\sigma}_2) = M_{k,1} + C_{\mathsf{Greedy}}(k, 1)$, while $C_{\mathsf{OPT}}(\boldsymbol{\sigma}_2) = \sum_{j=1}^{j=k} v^{(j)} \ - g(k) = P_{k,1}$. Therefore,

$$\frac{C_{\mathsf{OFF}}(\boldsymbol{\sigma}_2)}{C_{\mathsf{OPT}}(\boldsymbol{\sigma}_2)} = \frac{M_{k,1} + C_{\mathsf{Greedy}}(k,1)}{P_{k,1}}, = \mathsf{LCR}_k(1),$$
$$\geq \mathsf{LCR}_{i^\star(1)}(1).$$

since $\mathsf{LCR}_{i^\star(1)}(1)$ is the minimum LCR over all possible k. Thus, the optimum online algorithm OPT on input $\boldsymbol{\sigma}_2$ cannot have a competitive ratio strictly lower than $\mathsf{LCR}_{i^\star(1)}(1) = \mathsf{LCR}^{(1)}_{\boldsymbol{\sigma}_2}$. Thus, we get contradiction to hypothesis H_2 and equivalently to H_1. ◀

▶ Remark. The proof idea presented above is always an appealing candidate to show the optimality of any given online algorithm, however, fails most often. The key insight that makes it work for this problem is that the criteria used (minimize the LCR) by the min-LCR algorithm to select the number of packets to process at any time is an also an upper bound on the competitive ratio of the min-LCR algorithm. The typical methods to show lower bounds for online algorithms are: either by explicit construction of a lower bound example, or using Yao's min-max Lemma for randomized algorithms. The novelty with our approach is that the technique surprisingly works for any general convex energy cost function, and without having to explicitly evaluate the lower or upper bounds, which is very rarely found in literature.

After establishing that the min-LCR algorithm is an optimal online algorithm for any convex cost function g, we next concentrate on a specific cost function $g(k) = k^\alpha$ for $\alpha \geq 2$ that is the most popular choice in literature, to derive some concrete bounds on the competitive ratio of the min-LCR algorithm. Recall that min-LCR algorithm involves finding the number of jobs k that minimizes the LCR_k, which can be exhaustive. We next propose a simplified version of the min-LCR algorithm where exactly $k = \lfloor \beta m \rfloor$ or $k = \lceil \beta m \rceil$ jobs are processed in each slot depending on whichever one has lower LCR, and where β is the solution of the equation $x^\alpha + x^{\alpha-1} - 1 = 0$. We also consider the natural greedy special case of min-LCR algorithm that always processes m jobs, where m has been defined in the min-LCR algorithm.

We first derive a lower bound on the competitive ratio of any online algorithm, and next show that the simplified min-LCR algorithm (and consequently the min-LCR algorithm) achieves that lower bound for $\alpha = 2$. For $\alpha > 2$, we show that the competitive ratio of the simplified min-LCR algorithm is at most 3, while for $\alpha \geq 2.5$ it is at most 2.618.

4 $g(k) = k^\alpha$ for $\alpha \geq 2$

4.1 Lower Bound on the Competitive Ratio for $\alpha = 2$

▶ **Lemma 7.** *For any online algorithm* ALG, $\mathsf{r}_{\mathsf{ALG}} \geq \phi + 1$ *for cost function* $g(k) = k^\alpha$ *with* $\alpha = 2$, *where* $\phi = 1/\delta$ *and* $\delta = \frac{\sqrt{5}-1}{2}$.

Proof. We consider $2z$ jobs each with value $2z$ arriving at the start of slot 1. Thus, in slot 1 at most z jobs will be processed[1] by any algorithm since the cost function is $g(k) = k^2$, and the effective cost of processing job $z+1$ or higher is at least $g(z+1) - g(z) > 2z$. Let an online algorithm ALG choose to process k jobs out of the maximum possible z. Then we adversarially choose the deadlines of these k jobs to be infinity ∞, while keeping the deadlines of all other remaining jobs to be slot 1 itself. Since the offline optimal OFF knows the deadlines, it processes the maximum possible m jobs in slot 1, while processes the k jobs chosen by the ALG for slot 1, one at a time starting from slot 2 in k slots, making a profit of $2z \cdot z - z^2 + 2z \cdot k - k$, while the ALG makes only a profit of $2zk - k^2$. Thus, the competitive ratio of any online algorithm ALG as a function of k is

$$\mathsf{r}_{\mathsf{ALG}} \geq \min_k \frac{2z \cdot z - z^2 + 2z \cdot k - k}{2zk - k^2}.$$

[1] Processing any more jobs is unprofitable.

We take the limit as $z \to \infty$ to get that

$$\mathsf{r}_{\mathsf{ALG}} \geq \lim_{z\to\infty} \min_{k} \frac{2z \cdot z - z^2 + 2z \cdot k - k}{2zk - k^2} = \lim_{z\to\infty} \min_{k} \frac{2z \cdot z - z^2 + 2z \cdot k}{2zk - k^2},$$

since $\lim_{z\to\infty} \frac{-k}{2zk-k^2} = 0$ for any choice of k including the optimizer. It is easy to show that for $\delta = \frac{\sqrt{5}-1}{2}$, (see Appendix A.3)

$$\delta z = \arg\min_{k} \frac{2z \cdot z - z^2 + 2z \cdot k}{2zk - k^2}, \tag{8}$$

and $\lim_{m\to\infty} \frac{2z\cdot z - z^2 + 2z\cdot k}{2zk-k^2}|_{k=\frac{\sqrt{5}-1}{2}z} = \phi + 1$, where $\phi = \frac{1}{\delta}$. ◀

One can proceed similarly and get an expression for the lower bound on the competitive ratio for all values of $\alpha > 2$ as

$$\max_{z\in\mathbb{Z}^+} \max_{x\leq g(z+1)-2g(z)+g(z-1)} \min_{1\leq k\leq z} \left\{ \frac{[k(z^\alpha - (z-1)^\alpha + x - 1)] + [z(z^\alpha - (z-1)^\alpha + x) - z^\alpha]}{[k(z^\alpha - (z-1)^\alpha + x) - k^\alpha]} \right\}, \tag{9}$$

however, it is not easy to simplify it analytically, needing a numerical solution as presented in Fig. 1. To be more concrete, we next derive a slightly loose lower bound for $\alpha > 2$ as follows.

4.2 Lower Bound on the Competitive Ratio for $\alpha > 2$

▶ **Lemma 8.** *For any online algorithm* ALG, $\mathsf{r}_{\mathsf{ALG}} \geq \sqrt{2} + 1$ *for* $g(k) = k^\alpha$ *for all* $\alpha > 2$.

Proof. Consider the input where four jobs arrive at the beginning of slot 1 each with value $v = \left(1 + \frac{1}{\sqrt{2}}\right)[g(2) - \sqrt{2}g(1)]$. Any online algorithm ALG can process $k = 1$ or 2 or 3 or 4 jobs in slot 1. The choice of v is such that $v < g(3) - g(2)$, where $g(3) - g(2)$ is the effective cost of processing the third job in slot 1. Moreover, since g is a convex function $g(4) - g(3) > g(3) - g(2)$. Therefore processing the third or the fourth job in slot 1 incurs negative profit, and hence at most 2 jobs will be processed in slot 1.

Now depending on the choice of ALG for $k = 1, 2$, the adversarial choice of deadline will be that those k jobs will have deadline ∞, while the remaining $4 - k$ jobs will have deadline as slot 1 itself. A simple enumerative exercise reveals that $\mathsf{r}_{\mathsf{ALG}}|_{k=1} = \sqrt{2}+1, \mathsf{r}_{\mathsf{ALG}}|_{k=2} = \sqrt{2}+1$. The choice of v is the only non-trivial part in deriving this lower bound. ◀

4.3 Upper Bound on the Competitive Ratio for $\alpha \geq 2$

From here onwards we derive upper bounds on the competitive ratio of a simplified min-LCR algorithm, where we choose a particular number of jobs to process in every slot. The simplified min-LCR algorithm is as follows.

▶ **Lemma 9.** *The competitive ratio of sim-LCR Algorithm is at most* $\phi + 1$ *for* $\alpha = 2$ *or* $\alpha \geq 2.5$.

Combining Lemma 7 and Lemma 9, we get the following result.

▶ **Corollary 10.** *sim-LCR Algorithm is an optimal online algorithm for* $\alpha = 2$ *and the optimal competitive ratio for* $\alpha = 2$ *is* $\phi + 1$.

Algorithm 2: Simplified min-LCR algorithm (sim-LCR)

Input : Sequence $\boldsymbol{\sigma} = J_1, \ldots, J_n$

At slot τ, consider the union of all the non-processed jobs by the algorithm so far that have not-expired and the newly arriving jobs in slot τ and call it $E(\tau)$

Arrange the jobs in $E(\tau)$ in non-increasing order of their payoffs/value v

Let $E(\tau)^i = \{\text{first } i \text{ jobs in } E(\tau)\}$ and $E^{i-}(\tau) = E(\tau) \backslash E(\tau)^i$

Let $v^{(i)}$ be the value of job of $E(\tau)$ with the i^{th} highest value, $V^{(i)}$ be the sum of the values of the first i jobs with highest values

Let $m = \max_{\{v^{(j)} - [g(j) - g(j-1)] > 0\}} j$

Let $M_{i,\tau} = V^{(i)} - ig(1), P_{i,\tau} = V^{(i)} - g(i), C_{\mathsf{Greedy}}(i,\tau) = \max_{j \in E^{i-}(\tau)} \{V^{(j)} - g(j)\}$

Let β be the solution of $x^\alpha + x^{\alpha-1} - 1 = 0$

Compute $\mathsf{LCR}_i(\tau) = \dfrac{M_{i,\tau} + C_{\mathsf{Greedy}}(i,\tau)}{P_{i,\tau}}$ as in min-LCR algorithm for $i = \lfloor \beta m \rfloor$ or $i = \lceil \beta m \rceil$

Process either $i = \lfloor \beta m \rfloor$ or $i = \lceil \beta m \rceil$ jobs whichever one has lower LCR_i in slot τ.

Proof of Lemma 9. Let $i(\tau)$ be the optimizer among $i = \lfloor \beta m \rfloor$ or $i = \lceil \beta m \rceil$ that minimizes the LCR_i with the sim-LCR algorithm at slot τ. Then following an identical proof as for Lemma 5, it follows that the competitive ratio of the sim-LCR algorithm from (7),

$$\mathsf{r}_{\text{sim-LCR}}(\boldsymbol{\sigma}) = \frac{C_{\mathsf{OFF}}(\boldsymbol{\sigma})}{C_{\text{sim-LCR}}(\boldsymbol{\sigma})} \leq \frac{\sum_{\tau=1}^{\mathsf{last}} M_{i(\tau),\tau} + C_{\mathsf{Greedy}}(i(\tau),\tau)}{\sum_{\tau=1}^{\mathsf{last}} P_{i(\tau),\tau}} \leq \max_{\tau} \frac{M_{i(\tau),\tau} + C_{\mathsf{Greedy}}(i(\tau),\tau)}{P_{i(\tau),\tau}} = \max_{\tau} \mathsf{LCR}_{i(\tau)}(\tau). \tag{10}$$

Thus, to complete the proof, in the following, we show that the LCR_k for either $k = \lfloor \beta m \rfloor$ or $k = \lceil \beta m \rceil$ is less than $\phi + 1$ for the sim-LCR Algorithm at all slots τ. From the definition of the LCR

$$\mathsf{LCR}_k = \frac{M_{k,\tau} + C_{\mathsf{Greedy}}(k,\tau)}{P_{k,\tau}} \overset{(a)}{=\!\leq} \frac{\sum_{j=1}^{j=k} v^{(j)} \; - \; k \cdot g(1) + \sum_{j=1}^{j=m} v^{(j)} \; - \; g(m)}{\sum_{j=1}^{j=k} \; v^{(j)} \; - \; g(k)}, \tag{11}$$

(a) follows from the definition of $M_{k,\tau}$ and an upper bound derived for $C_{\mathsf{Greedy}}(k,\tau)$ in Appendix A.2. Recall that the set of available jobs $E(\tau)$ at each slot τ are arranged in decreasing order of their values, hence, for $k \leq m$, the average of the values of the first k jobs is greater than the average of the values of the first m. Hence, we have $\frac{m}{k} \sum_{j=1}^{j=k} v^{(j)} \geq \sum_{j=1}^{j=m} v^{(j)}$. Substituting this in (11) yields

$$\begin{aligned} \mathsf{LCR}_k &\leq \frac{\sum_{j=1}^{j=k} v^{(j)} \; - \; k + \frac{m}{k}\left(\sum_{j=1}^{j=k} v^{(j)}\right) \; - \; g(m)}{\sum_{j=1}^{j=k} \; v^{(j)} \; - \; g(k)} \\ &= \frac{m}{k} + 1 + \frac{g(k) + \frac{m}{k} g(k) - g(m) - k}{\sum_{j=1}^{j=k} \; v^{(j)} \; - \; g(k)}. \end{aligned} \tag{12}$$

Consider the numerator of the second term $f(k) = g(k) + \frac{m}{k} g(k) - g(m) - k$ at $k = \beta m$. By definition, $f(\beta m) = m^\alpha(\beta^\alpha + \beta^{\alpha-1} - 1) - \beta m \leq 0$ since β is the solution of the equation $x^\alpha + x^{\alpha-1} - 1 = 0$. Moreover, $f(\lfloor \beta m \rfloor) = (\lfloor \beta m \rfloor)^\alpha + \frac{m}{(\lfloor \beta m \rfloor)} (\lfloor \beta m \rfloor)^\alpha - m^\alpha - \lfloor \beta m \rfloor \leq 0$, since $(\lfloor \beta m \rfloor)^\alpha + \frac{m}{(\lfloor \beta m \rfloor)} (\lfloor \beta m \rfloor)^\alpha - m^\alpha \leq m^\alpha(\beta^\alpha + \beta^{\alpha-1} - 1)$. Using this in (12), we get

$$\mathsf{LCR}_{\lfloor \beta m \rfloor} \leq \frac{m}{k} + 1. \tag{13}$$

For $\alpha \geq 2.5$, by direct computation, one can check that for $m = \{1, 3, 6, 8, 9, 10, 11, 12\}$ and $\forall m \geq 13, \mathsf{LCR}_{\lfloor \beta m \rfloor} \leq \phi + 1$. For the remaining values of $m = 2, 4, 5, 7$ either $\mathsf{LCR}_{\lfloor \beta m \rfloor} \leq \phi + 1$ or $\mathsf{LCR}_{\lceil \beta m \rceil} \leq \phi + 1$, as derived in Appendix A.4.

For $\alpha = 2$, a little more involved approach is needed as follows. For $\alpha = 2$, $\beta = \delta$, where β is the solution of the equation $x^\alpha + x^{\alpha-1} - 1 = 0$. Let γ be the positive root of the equation $x^2 + (m-1)x - m^2 = 0$. From (11), using $\alpha = 2$,

$$\mathsf{LCR}_k = \frac{m}{k} + 1 + \frac{k^2 + mk - m^2 - k}{\sum_{j=1}^{j=k} v^{(j)} - k^2}. \tag{14}$$

The quadratic equation $k^2 + mk - m^2 - k$ is increasing in the interval $[\delta m, \gamma]$ ($\delta = \frac{\sqrt{5}-1}{2}$) with value < 0 at $k = \delta m$ and $= 0$ at $k = \gamma$ (by definition of γ). So at all intermediate values of $k \in (\delta m, \gamma)$, the value of the equation must be less than 0. Thus, if $\lceil \delta m \rceil \in [\delta m, \gamma]$, we have $\mathsf{LCR}_{\lceil \delta m \rceil} \leq \frac{m}{\lceil \delta m \rceil} + 1 \leq \frac{m}{\delta m} + 1 = \phi + 1$.

Next, we consider the case when $\lceil \delta m \rceil \in (\gamma, \delta m + 1]$. The numerator of the third term in the RHS of (14) is an increasing function for $(\gamma, \delta m + 1]$ from the definition of γ. Thus, since $\sum_{j=1}^{j=k} v^{(j)} \geq k[g(m) - g(m-1)]$ where $v^{(j)}$ is the j^{th} highest value of the job when arranged in a non-increasing order of values. Thus, from (14), we get

$$\mathsf{LCR}_k \leq \frac{m}{k} + 1 + \frac{k^2 + mk - m^2 - k}{(2m-1)k - k^2}. \tag{15}$$

Let $\psi(k) = \frac{m}{k} + 1 + \frac{k^2 + mk - m^2 - k}{(2m-1)k - k^2}$, then it follows that $\frac{\partial \psi(k)}{\partial k} \geq 0$ for $k \in (\gamma, \delta m + 1]$, and $\psi(\delta m + 1) \leq \phi + 1$ for $m \geq 5$. This implies that $\mathsf{LCR}_k \leq \phi + 1$ for all $k \in (\gamma, \delta m + 1]$. Thus, even if $\lceil \delta m \rceil \in (\gamma, \delta m + 1]$, $\mathsf{LCR}_{\lceil \beta m \rceil} \leq \phi + 1$ for $m \geq 5$. For $m = 1$, $\mathsf{LCR}_{\lceil \delta m \rceil} \leq 2$, while for

$$m = 3 \implies LCR_{\lceil \delta m \rceil} \leq \frac{3}{2} + 1 + \frac{2^2 + 3 \cdot 2 - 3^2 - 2}{(2 \cdot 3 - 1)2 - 2^2} < \phi + 1 \tag{16}$$

$$m = 4 \implies LCR_{\lceil \delta m \rceil} \leq \frac{4}{3} + 1 + \frac{3^2 + 4 \cdot 3 - 4^2 - 3}{(2 \cdot 4 - 1)3 - 3^2} < \phi + 1. \tag{17}$$

For $m = 2$, the proof is identical to the one for the case of $\alpha \geq 2.5$ for $m = 2$ as provided in Appendix A.4. ◀

4.4 Upper Bound on the Competitive Ratio of the min-LCR algorithm for $\alpha \geq 2$

We can obtain a slightly loose upper bound for all values of $\alpha \geq 2$ by enforcing the min-LCR algorithm to process all m jobs, where m is largest number of jobs that can be processed at any slot with positive profit for each job. This restriction can essentially be seen as a greedy analogue of the min-LCR algorithm, and we call it Greedy.

▶ **Lemma 11.** *The competitive ratio of the* Greedy *algorithm (min-LCR algorithm with $i^\star = m$) is at most* 3 *for all $\alpha > 2$.*

Proof can be found in Appendix A.5.

References

1 Susanne Albers, Fabian Müller, and Swen Schmelzer. Speed scaling on parallel processors. *Algorithmica*, 68(2):404–425, 2014.

2 Yossi Azar, Nikhil R Devanur, Zhiyi Huang, and Debmalya Panigrahi. Speed scaling in the non-clairvoyant model. In *Proceedings of the 27th ACM symposium on Parallelism in Algorithms and Architectures*, pages 133–142. ACM, 2015.

3 Yossi Azar, Bala Kalyanasundaram, Serge Plotkin, Kirk R Pruhs, and Orli Waarts. Online load balancing of temporary tasks. In *Workshop on algorithms and data structures*, pages 119–130. Springer, 1993.

4 Nikhil Bansal, Ho-Leung Chan, Tak-Wah Lam, and Lap-Kei Lee. Scheduling for speed bounded processors. In *International Colloquium on Automata, Languages, and Programming*, pages 409–420. Springer, 2008.

5 Nikhil Bansal, Ho-Leung Chan, Kirk Pruhs, and Dmitriy Katz. Improved bounds for speed scaling in devices obeying the cube-root rule. *Automata, languages and programming*, pages 144–155, 2009.

6 Nikhil Bansal, Tracy Kimbrel, and Kirk Pruhs. Speed scaling to manage energy and temperature. *Journal of the ACM (JACM)*, 54(1):3, 2007.

7 Nikhil Bansal, Kirk Pruhs, and Cliff Stein. Speed scaling for weighted flow time. *SIAM Journal on Computing*, 39(4):1294–1308, 2009.

8 Neal Barcelo, Peter Kling, Michael Nugent, and Kirk Pruhs. *Optimal Speed Scaling with a Solar Cell*, pages 521–535. Springer International Publishing, Cham, 2016. doi:10.1007/978-3-319-48749-6_38.

9 Sanjoy Baruah, Gilad Koren, Bhubaneswar Mishra, Arvind Raghunathan, Louis Rosier, and Dennis Shasha. On-line scheduling in the presence of overload. In *Foundations of Computer Science, 1991. Proceedings., 32nd Annual Symposium on*, pages 100–110. IEEE, 1991.

10 Ho-Leung Chan, Joseph Wun-Tat Chan, Tak-Wah Lam, Lap-Kei Lee, Kin-Sum Mak, and Prudence WH Wong. Optimizing throughput and energy in online deadline scheduling. *ACM Transactions on Algorithms (TALG)*, 6(1):10, 2009.

11 Ho-Leung Chan, Jeff Edmonds, Tak-Wah Lam, Lap-Kei Lee, Alberto Marchetti-Spaccamela, and Kirk Pruhs. Nonclairvoyant speed scaling for flow and energy. *Algorithmica*, 61(3):507–517, 2011.

12 Aaron Coté, Adam Meyerson, Alan Roytman, Michael Shindler, and Brian Tagiku. Energy-efficient online scheduling with deadlines. *unpublished manuscript*, 2010.

13 Anupam Gupta, Ravishankar Krishnaswamy, and Kirk Pruhs. Nonclairvoyantly scheduling power-heterogeneous processors. *Sustainable Computing: Informatics and Systems*, 1(3):248–255, 2011.

14 Xin Han, Tak-Wah Lam, Lap-Kei Lee, Isaac KK To, and Prudence WH Wong. Deadline scheduling and power management for speed bounded processors. *Theoretical Computer Science*, 411(40-42):3587–3600, 2010.

15 Sandy Irani, Sandeep Shukla, and Rajesh Gupta. Algorithms for power savings. *ACM Transactions on Algorithms (TALG)*, 3(4):41, 2007.

16 Tak-Wah Lam, Lap-Kei Lee, Isaac KK To, and Prudence WH Wong. Speed scaling functions for flow time scheduling based on active job count. In *European Symposium on Algorithms*, pages 647–659. Springer, 2008.

17 Marco Riedel. Online request server matching. *Theoretical computer science*, 268(1):145–160, 2001.

18 Adam Wierman, Lachlan LH Andrew, and Ao Tang. Power-aware speed scaling in processor sharing systems. In *INFOCOM 2009, IEEE*, pages 2007–2015. IEEE, 2009.

19 Frances Yao, Alan Demers, and Scott Shenker. A scheduling model for reduced cpu energy. In *Foundations of Computer Science, 1995. Proceedings., 36th Annual Symposium on*, pages 374–382. IEEE, 1995.

A Appendix

A.1 Numerical Evaluation of the Lower bound (9)

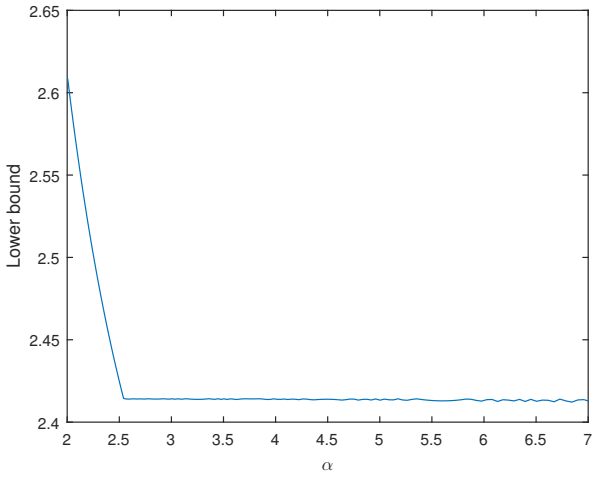


Figure 1 Numerical Evaluation of the Lower bound (9) as a function of α.

A.2 Proof for upper bounding $C_{\mathsf{Greedy}}(k,\tau)$ in (11)

▶ **Lemma 12.**

$$C_{\mathsf{Greedy}}(k,\tau) \leq \sum_{j=1}^{j=m} v^{(j)} - g(m).$$

Proof. Recall that the effective cost of the i^{th} job is $c_i = g(i) - g(i-1)$.

$$\begin{aligned} C_{\mathsf{Greedy}}(k,\tau) &\overset{(a)}{=} \max_{j \in E^{k-}(\tau)} (V^{(j)} - g(j)), \\ &\overset{(b)}{\leq} \max_{j \in E(\tau)} (V^{(j)} - g(j)), \\ &\overset{(c)}{=} (V^{(j^\star)} - g(j^\star)) = \sum_{i=1}^{j^\star} v^{(i)} - c_i, \\ &\overset{(d)}{=} \sum_{j=1}^{j=m} v^{(j)} - g(m), \end{aligned}$$

(a) follows from the definition, (b) follows since $E^{k-}(\tau) \subseteq E(\tau)$, (c) follows by defining the optimizer j in (b) to be $j^\star$, while (d) is true because of the following inequalities

$$\forall i > m: \quad v^{(i)} \overset{c}{\leq} v^{(m+1)} \overset{d}{\leq} c_{m+1} \overset{e}{\leq} c_i \implies [v^{(i)} - c_i] \leq 0$$

$$\forall i \le m: \quad v^{(i)} \overset{c}{\ge} v^{(m)} \overset{d}{\ge} c_m \overset{e}{\ge} c_i \implies [v^{(i)} - c_i] \ge 0,$$

where (c) follows since jobs are indexed in the non-increasing order of their values, and (d) follows from the definition of m, and (e) follows from the convexity of $g(k)$. ◀

A.3 Proof of (8)

▶ **Lemma 13.**

$$\arg\min_k \left\{ \frac{z^2 + 2zk}{2zk - k^2} \right\} = \delta z.$$

Proof. Taking the derivative, we get

$$\frac{\partial}{\partial k}\left(\frac{2zk + z^2}{2zk - k^2}\right) = \frac{2z(k^2 + zk - z^2)}{(2zk - k^2)^2}, \tag{18}$$

which when equated to zero, we get $k^\star = \delta z$. Taking the second derivative and evaluating at δm, we show that $k^\star = \delta m$ is a local minima as follows.

$$\frac{\partial^2}{\partial k^2}\left(\frac{2zk + z^2}{2zk - k^2}\right) = \frac{[2z(2zk - k^2)]\,[2z^3 - 6z^2k + 4z^3 + 3zk^2]}{(2zk - k^2)^4},$$

$$\implies \frac{\partial^2}{\partial k^2}\left(\frac{2zk + z^2}{2zk - k^2}\right)\Bigg|_{k=\delta z} \overset{(a)}{>} 0,$$

where (a) follows since

$$[2k^3 - 6z^2k + 4z^3 + 3zk^2]\Big|_{k=\delta z} = [2z^3(2\delta^3 + 3\delta^2 - 6\delta + 4)] > 0.$$

Evaluating at the boundary points of $k = 1$, we get $\frac{z^2 + 2z}{2z - 1} > \phi + 1$ and at $k = z$, $\frac{z^2 + 2z^2}{2z^2 - z^2} = 3 \ge \phi + 1$. Thus, we conclude that $k = \delta z$ is the minima. ◀

A.4 Proof of bounding $\mathsf{LCR}_{\lfloor \beta m \rfloor}$ and $\mathsf{LCR}_{\lceil \beta m \rceil}$ for $m = 2, 4, 5, 7$

▶ **Lemma 14.** *For $m = \{2, 4, 5, 7\}$ $\forall \alpha \ge 2.5$ either $\mathsf{LCR}_{\lfloor \beta m \rfloor}$ or $\mathsf{LCR}_{\lceil \beta m \rceil}$ is $\le \phi + 1$.*

Proof. $m = 2$: $\alpha \ge 2 \implies \lfloor \beta m \rfloor = 1$ and $\lceil \beta m \rceil = 2$. When $v^{(2)} \le \left(1 + \frac{1}{\sqrt{2}}\right)[g(2) - \sqrt{2}g(1)]$,

$$\mathsf{LCR}_1 \overset{p}{=} \frac{[v^{(1)} - g(1)] + [v^{(2)} + v^{(3)} - g(2)]}{[v^{(1)} - g(1)]} \overset{q}{\le} 1 + \frac{2v^{(2)} - g(2)}{v^{(2)} - g(1)} \overset{r}{\le} \phi + 1,$$

while when $v^{(2)} \ge \left(1 + \frac{1}{\sqrt{2}}\right)[g(2) - \sqrt{2}g(1)]$,

$$\mathsf{LCR}_2 \overset{p}{=} \frac{[v^{(1)} + v^{(2)} - 2g(1)] + [v^{(3)} + v^{(4)} - g(2)]}{[v^{(1)} + v^{(2)} - g(2)]} \overset{q}{\le} 1 + \frac{2v^{(2)} - 2g(1)}{2v^{(2)} - g(2)} \overset{r}{\le} \phi + 1,$$

where p follows from the definition of LCR, q follows from the fact that $v^{(1)} \ge v^{(2)} \ge v^{(3)} \ge v^{(4)}$ and r follows by simple evaluation in the appropriate range of choice for $v^{(2)}$. $m = 4$: When $\alpha \in [2.5, 2.945] \implies \lceil \beta m \rceil = \lceil \beta * 4 \rceil = 3$

$$\implies \mathsf{LCR}_{\lceil \beta m \rceil} \overset{a}{=} \frac{4}{3} + 1 + \frac{7 \cdot 3^{(\alpha-1)} - 4 \cdot 4^{(\alpha-1)}}{12(4^{(\alpha-1)} - 3^{(\alpha-1)})} \overset{b}{\le} 2 + \frac{1}{4\left(\frac{4^{(\alpha-1)}}{3^{(\alpha-1)}} - 1\right)} \overset{c}{\le} \phi + 1,$$

$$\alpha \geq 2.945 \implies \lfloor \beta m \rfloor \geq 3 \implies \mathsf{LCR}_{\lfloor \beta m \rfloor} \overset{d}{\leq} \frac{m}{k} + 1 = \frac{4}{3} + 1 \leq \phi + 1.$$

$m = 5$: When $\alpha \in [2.5, 3.641] \implies \lceil \beta m \rceil = 4$

$$\implies \mathsf{LCR}_{\lceil \beta m \rceil} \overset{a}{\leq} \frac{5}{4} + 1 + \frac{9 \cdot 4^{(\alpha-1)} - 5 \cdot 5^{(\alpha-1)}}{20(5^{(\alpha-1)} - 4^{(\alpha-1)})} \overset{b}{=} 2 + \frac{1}{5\left(\frac{5^{(\alpha-1)}}{4^{(\alpha-1)}} - 1\right)} \overset{e}{\leq} \phi + 1,$$

$$\alpha \geq 3.641 \implies \lfloor \beta m \rfloor \geq 4 \implies \mathsf{LCR}_{\lfloor \beta m \rfloor} \overset{d}{\leq} \frac{m}{k} + 1 = \frac{5}{4} + 1 \leq \phi + 1.$$

$m = 7$:

$$\alpha \in [2.5, 2.6] \implies \lceil \beta m \rceil = 5 \implies \mathsf{LCR}_{\lceil \beta m \rceil} \overset{a}{\leq} \frac{7}{5} + 1 + \frac{\frac{12}{5}5^{\alpha} - 7^{\alpha} - 5}{5[7^{\alpha} - 6^{\alpha} - 5^{\alpha}]} \leq \phi + 1,$$

$$\alpha \geq 2.6 \implies \lfloor \beta m \rfloor \geq 5 \implies \mathsf{LCR}_{\lfloor \beta m \rfloor} \overset{d}{\leq} \frac{m}{k} + 1 = \frac{7}{5} + 1 \leq \phi + 1.$$

In all the cases, a follows from (12) along with the fact that $\sum_{j=1}^{j=k} v^{(j)} \geq k[g(m) - g(m-1)]$ and $g(k) = k^{\alpha}$, (b) follows by re-arranging terms, c is true because $\forall\ \alpha \geq 2.5$ we have $\left(\frac{4^{(\alpha-1)}}{3^{(\alpha-1)}} - 1\right) - 1 \geq 0.53$, d follows from (13), and e is true because $\forall \alpha \geq 2.5$ we have $\left(\frac{5^{(\alpha-1)}}{4^{(\alpha-1)}} - 1\right) \geq 0.39$. ◀

A.5 Proof of Lemma 11

Proof. From (10), $\mathsf{r}_{\text{sim-LCR}}(\boldsymbol{\sigma}) \leq \max_{\tau} \mathsf{LCR}_{i(\tau)}$. In this greedy case, $i(\tau) = m$, always, and hence to prove the result we show that $\mathsf{LCR}_m \leq 3$ for all slot τ and all inputs $\boldsymbol{\sigma}$. From (11), we have that $\mathsf{LCR}_k \leq \frac{m}{k} + 1 + \frac{g(k) + \frac{m}{k}g(k) - g(m) - k}{\sum_{j=1}^{j=k} v^{(j)} - g(k)}$. Choosing $k = m$ and substituting it in the above equation,

$$\begin{aligned}
\mathsf{LCR}_m &\leq \frac{m}{m} + 1 + \frac{g(m) + \frac{m}{m}g(m) - g(m) - m}{\sum_{j=1}^{j=m} v^{(j)} - g(m)}, \\
&\leq 1 + 1 + \frac{g(m) - m}{\sum_{j=1}^{j=m} v^{(j)} - g(m)}, \\
&\overset{(a)}{=} 1 + \frac{\sum_{j=1}^{j=m} v^{(j)} - m}{\sum_{j=1}^{j=m} v^{(j)} - g(m)}, \\
&= 1 + \frac{1}{\frac{\sum_{j=1}^{j=m} v^{(j)} - g(m)}{\sum_{j=1}^{j=m} v^{(j)} - m}}, \\
&= 1 + \frac{1}{1 - \frac{g(m) - m}{\sum_{j=1}^{j=m} v^{(j)} - m}}, \\
&\overset{(b)}{=} 1 + \frac{1}{1 - h(m, g)},
\end{aligned}$$

where (a) follows by adding the second and third terms, and (b) by defining of $h(m,g) = \frac{g(m)-m}{\sum_{j=1}^{j=m} v^{(j)} - m}$.

Next, we prove that $h(m,g) \leq 1/2$, for $\alpha \geq 2$, from which it follows that $\mathsf{LCR}_m \leq 3$. Thus, the competitive ratio of the min-LCR algorithm is at most 3 for all $\alpha > 2$.

From the definition of m, $m = \max_{\{v^{(j)} - [g(j)-g(j-1)]>0\}} j$, all the m highest valued jobs have a value greater than or equal to the effective cost of processing them in slot m, i.e., $v_k \geq c_k = v(k) - v(k-1)$ for $k \leq m$, and hence $\sum_{j=1}^{m} v^{(j)} \geq m[g(m) - g(m-1)]$, thus,

$$h(m,g) \leq \frac{g(m)-m}{m[g(m)-g(m-1)] - m}.$$

Let $\Theta(\alpha) = \dfrac{g(m)-m}{m[g(m)-g(m-1)] - m} = \dfrac{m^\alpha - m}{m[m^\alpha - (m-1)^\alpha] - m}$, since $g(k) = k^\alpha$.

▶ **Lemma 15.** $\forall \alpha \geq 2 \quad \dfrac{\partial \Theta(\alpha)}{\partial \alpha} \leq 0.$

Proof. Taking the derivative and rearranging the terms we get

$$\begin{aligned}
\frac{\partial \Theta(\alpha)}{\partial \alpha} &= -\frac{m^2(m-1)[m^{\alpha-1}((m-1)^{\alpha-1}-1)\log(m) - (m-1)^{\alpha-1}(m^{\alpha-1}-1)\log(m-1)]}{(m[m^\alpha - (m-1)^\alpha] - m)^2}, \\
&\overset{(a)}{=} -\left[\frac{(m^2)(m-1)((m-1)^{\alpha-1}-1)(m^{\alpha-1}-1)}{(\alpha-1)(m[m^\alpha-(m-1)^\alpha]-m)^2}\right] \\
&\quad \cdot \left[\frac{(m^{\alpha-1})\log(m^{\alpha-1})}{(m^{\alpha-1}-1)} - \frac{((m-1)^{\alpha-1})\log((m-1)^{\alpha-1})}{((m-1)^{\alpha-1}-1)}\right], \\
&\overset{(b)}{\leq} 0,
\end{aligned}$$

where (a) follows from writing $\log(m) = \dfrac{\log(m^{\alpha-1})}{\alpha-1}$, and (b) follows from the following three observations i) $\dfrac{x\log(x)}{x-1}$ is an increasing function, ii) $m^{\alpha-1} \geq (m-1)^{\alpha-1}$ and iii) the first term is positive. Note that $\log(m-1)$ is not defined for $m = 1$ but for $m = 1$ observe that $\Theta(\alpha)$ is always 0. Therefore $\forall \alpha \geq 2$ $h(m,g) \leq 0.5$, since for $\alpha = 2$, $\Theta(\alpha) = \dfrac{m^2 - m}{(m^2 - (m-1)^2) - m)} = 0.5$. ◀

◀

Multi-Agent Submodular Optimization

Richard Santiago
McGill University, Montreal, Canada
richard.santiagotorres@mail.mcgill.ca

F. Bruce Shepherd
University of British Columbia, Vancouver, Canada
fbrucesh@cs.ubc.ca

Abstract

Recent years have seen many algorithmic advances in the area of submodular optimization: (SO) $\min / \max \ f(S) : S \in \mathcal{F}$, where $\mathcal{F}$ is a given family of feasible sets over a ground set V and $f : 2^V \to \mathbb{R}$ is submodular. This progress has been coupled with a wealth of new applications for these models. Our focus is on a more general class of *multi-agent submodular optimization* (MASO) $\min / \max \sum_{i=1}^{k} f_i(S_i) : S_1 \uplus S_2 \uplus \cdots \uplus S_k \in \mathcal{F}$. Here we use $\uplus$ to denote disjoint union and hence this model is attractive where resources are being allocated across k agents, each with its own submodular cost function $f_i()$. This was introduced in the minimization setting by Goel et al. In this paper we explore the extent to which the approximability of the multi-agent problems are linked to their single-agent versions, referred to informally as the *multi-agent gap*.

We present different reductions that transform a multi-agent problem into a single-agent one. For minimization, we show that (MASO) has an $O(\alpha \cdot \min\{k, \log^2(n)\})$-approximation whenever (SO) admits an α-approximation over the convex formulation. In addition, we discuss the class of "bounded blocker" families where there is a provably tight $O(\log n)$ multi-agent gap between (MASO) and (SO). For maximization, we show that monotone (resp. nonmonotone) (MASO) admits an $\alpha(1 - 1/e)$ (resp. $\alpha \cdot 0.385$) approximation whenever monotone (resp. nonmonotone) (SO) admits an α-approximation over the multilinear formulation; and the $1 - 1/e$ multi-agent gap for monotone objectives is tight. We also discuss several families (such as spanning trees, matroids, and p-systems) that have an (optimal) multi-agent gap of 1. These results substantially expand the family of tractable models for submodular maximization.

2012 ACM Subject Classification Theory of computation → Submodular optimization and polymatroids

Keywords and phrases submodular optimization, multi-agent, approximation algorithms

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.23

Related Version The full version of this work can be found in [35], `https://arxiv.org/abs/1803.03767`.

Acknowledgements We thank Chandra Chekuri for valuable comments and suggestions.

1 Introduction

A set function $f : 2^V \to \mathbb{R}$ is *submodular* if $f(S) + f(T) \geq f(S \cup T) + f(S \cap T)$ for any $S, T \subseteq V$. We say that f is *monotone* if $f(S) \leq f(T)$ whenever $S \subseteq T$. Throughout, we usually assume that our functions are nonnegative and satisfy $f(\emptyset) = 0$. We work in the *value oracle model*, where for a given set S we can query the oracle to find its value $f(S)$.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018).
Editors: Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer; Article No. 23; pp. 23:1–23:20

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

For a family of feasible sets $\mathcal{F} \subseteq 2^V$ on a finite ground set V we consider the following broad class of submodular optimization (SO) problems:

$$\text{SO}(\mathcal{F}) \qquad \text{Min} \;/\; \text{Max} \;\; f(S) : S \in \mathcal{F} \tag{1}$$

where f is a nonnegative submodular set function on V. There has been an impressive recent stream of activity around these problems for a variety of set families $\mathcal{F}$. We explore the connections between these (single-agent) problems and their multi-agent incarnations. In the *multi-agent (MA)* version, we have k agents each of which has an associated nonnegative submodular set function f_i, $i \in [k]$. As before, we are looking for sets $S \in \mathcal{F}$, however, we now have a 2-phase task: the elements of S must also be partitioned amongst the agents. Hence we have set variables S_i and seek to optimize $\sum_i f_i(S_i)$. This leads to the multi-agent submodular optimization (MASO) versions:

$$\text{MASO}(\mathcal{F}) \qquad \text{Min} \;/\; \text{Max} \;\; \textstyle\sum_{i=1}^{k} f_i(S_i) : S_1 \uplus S_2 \uplus \cdots \uplus S_k \in \mathcal{F}. \tag{2}$$

The special case when $\mathcal{F} = \{V\}$ has been previously examined both for minimization (the minimum submodular cost allocation problem [17, 39, 7, 5]) and maximization (submodular welfare problem [29, 40]). For general families $\mathcal{F}$, however, we are only aware of the development in Goel et al. [12] for the minimization setting. A natural first question is whether any multi-agent problem could be directly reduced (or encoded) to a single-agent one over the same ground set V. Goel et al. give an explicit example where such a reduction does not exist. More emphatically, they show that when $\mathcal{F}$ consists of vertex covers in a graph, the *single-agent (SA)* version (i.e., (1)) has a 2-approximation while the MA version has an inapproximability lower bound of $\Omega(\log n)$.

Our first main objective is to explain the extent to which approximability for multi-agent problems is intrinsically connected to their single-agent versions, which we also refer to as the *primitive* associated with $\mathcal{F}$. We refer to the *multi-agent (MA) gap* as the approximation-factor loss incurred by moving to the MA setting.

Our second objective is to extend the multi-agent model and show that in some cases this larger class remains tractable. Specifically, we define the *capacitated multi-agent submodular optimization (CMASO) problem* as follows:

$$\text{CMASO}(\mathcal{F}) \qquad \begin{array}{rl} \max / \min & \sum_{i=1}^{k} f_i(S_i) \\ \text{s.t.} & S_1 \uplus \cdots \uplus S_k \in \mathcal{F} \\ & S_i \in \mathcal{F}_i \,,\; \forall i \in [k] \end{array} \tag{3}$$

where we are supplied with subfamilies $\mathcal{F}_i$. Many existing applications fit into this framework and some of these can be enriched through the added flexibility of the capacitated model. We illustrate this with concrete examples in the next section.

Prior work in both the single and multi-agent settings is summarized in Section 1.2. Our main contributions are presented in Section 1.3.

1.1 Some applications of (capacitated) multi-agent optimization

In this section we present several problems in the literature which are special cases of Problem (2) and the more general Problem (3). We also indicate how the extra generality of CMASO (i.e. (3)) gives modelling advantages. We start with the maximization setting.

▶ **Example 1** (The Submodular Welfare Problem)**.** The most basic problem in the maximization setting arises when we take the family $\mathcal{F} = \{V\}$. This describes a well-known model (introduced in [29]) for allocating goods to agents, each of which has a monotone submodular

valuation (utility) function over baskets of goods. This is formulated as (2) by considering nonnegative monotone functions f_i and $\mathcal{F} = \{V\}$. The CMASO framework allows us to incorporate additional constraints by defining the families $\mathcal{F}_i$ appropriately. For instance, one can impose cardinality constraints on the number of elements that an agent can take, or to only allow agent i to take a set S_i of elements satisfying some bounds $L_i \subseteq S_i \subseteq U_i$.

▶ **Example 2** (The Separable Assignment Problem)**.** An instance of the Separable Assignment Problem (SAP) consists of m items and n bins. Each bin j has an associated downwards closed collection of feasible sets $\mathcal{F}_j$, and a modular function $v_j(i)$ that denotes the value of placing item i in bin j. The goal is to choose disjoint feasible sets $S_j \in \mathcal{F}_j$ so as to maximize $\sum_{j=1}^{n} v_j(S_j)$. This well-studied problem ([11, 14, 4]) corresponds to a CMASO instance with modular objectives, $\mathcal{F} = 2^V$, and downwards closed families $\mathcal{F}_i$.

▶ **Example 3** (Sensor Placement)**.** The problem of placing sensors and information gathering has been popular in the submodularity literature [24, 26, 25]. We are given a set of sensors V and a set of possible locations $\{1, 2, \ldots, k\}$ where the sensors can be placed. There is also a budget constraint restricting the number of sensors that can be deployed. The goal is to place sensors at some of the locations so as to maximize the total "informativeness". Consider a multi-agent objective function $\sum_{i\in[k]} f_i(S_i)$, where $f_i(S_i)$ measures the informativeness of placing sensors S_i at location i. It is then natural to consider a diminishing return (i.e. submodularity) property for the f_i. We can formulate this problem as MASO where $\mathcal{F} := \{S \subseteq V : |S| \leq b\}$ imposes the budget constraint. We can also use CMASO for additional modelling flexibility. For instance, we may define $\mathcal{F}_i = \{S \subseteq V_i : |S| \leq b_i\}$ where V_i are the allowed sensors for location i and b_i an upper bound on the sensors located there.

We now discuss applications of MASO and CMASO in the minimization setting.

▶ **Example 4** (Minimum Submodular Cost Allocation)**.** The most basic problem in the minimization setting arises when we simply take $\mathcal{F} = \{V\}$. This problem, $\min \sum_{i=1}^{k} f_i(S_i) : S_1 \uplus \cdots \uplus S_k = V$, has been widely considered in the literature for both monotone [39] and nonmonotone functions [5, 7], and is referred to as the Minimum Submodular Cost Allocation (MSCA) problem[1] (introduced in [17, 39] and further developed in [5]). This is formulated as (2) by taking $\mathcal{F} = \{V\}$. The CMASO framework allows us to incorporate additional constraints into this problem. The most natural are to impose cardinality constraints on the number of elements that an agent can take, or to only allow agent i to take a set S_i of elements satisfying some bounds $L_i \subseteq S_i \subseteq U_i$.

▶ **Example 5** (Multi-agent Minimization)**.** Goel et al [12] consider the special cases of MASO($\mathcal{F}$) where the objectives are nonnegative monotone submodular and $\mathcal{F}$ is either the family of vertex covers, spanning trees, perfect matchings, or shortest st paths.

1.2 Related work

Single Agent Optimization. Minimizing a submodular function is a classical problem which can be solved in polytime [15, 36, 19]. Unconstrained maximization on the other hand is known to be inapproximable for general submodular functions but admits a polytime constant-factor approximation algorithm when f is nonnegative [2, 8].

For constrained maximization, the classical work [32, 33, 10] established an optimal $(1 - 1/e)$-approximation for nonnegative monotone maximization under a cardinality constraint, and a $(1/(k+1))$-approximation under k matroid constraints. The latter is almost tight since

[1] Sometimes referred to as submodular procurement auctions.

there is an $\Omega(\log(k)/k)$ inapproximability result [18]. For nonnegative monotone functions, [40, 4] give an optimal $(1-1/e)$-approximation based on the multilinear extension when $\mathcal{F}$ is a matroid; and [28] gives a local-search algorithm that achieves a $(1/k-\epsilon)$-approximation (for any fixed $\epsilon > 0$) when $\mathcal{F}$ is a k-matroid intersection. For nonnegative nonmonotone functions, a 0.385-approximation ([1]) is the best factor known for a matroid constraint. In [27] a $1/(k+O(1))$-approximation is given for k matroid constraints with k fixed, and [16] gives a simple "multi-greedy" algorithm that matches the approximation of Lee et al. but is polytime for any k. Finally, Chekuri et al [42] introduce a general framework based on relaxation-and-rounding that allows for combining different types of constraints.

For constrained minimization the news are worse [12, 38, 20]. If $\mathcal{F}$ consists of spanning trees Goel et al [12] show a lower bound of $\Omega(n)$, while if $\mathcal{F}$ corresponds to the cardinality constraint $\{S : |S| \geq k\}$ Svitkina and Fleischer [38] show a lower bound of $\tilde{\Omega}(\sqrt{n})$. There are a few exceptions. The problem can be solved exactly when $\mathcal{F}$ is a ring family ([36]), triple family ([15]), or parity family ([13]). In the context of NP-Hard problems, there are almost no cases where good approximations exist. We have a 2-approximation ([12, 20]) for submodular vertex cover, and an $O(k)$-approximation for k-uniform hitting set.

Multi-agent Problems. In the maximization side, the main studied problem is Submodular Welfare Maximization ($\mathcal{F} = \{V\}$) for which the initial 1/2-approximation [29] was improved to $1-1/e$ by Vondrak [40]. This approximation is in fact optimal [22, 31]. We are not aware of any maximization work for MASO($\mathcal{F}$) for nontrivial families $\mathcal{F}$.

For multi-agent minimization, MSCA (i.e. $\mathcal{F} = \{V\}$) is the most studied application of MASO($\mathcal{F}$). For nonnegative monotone functions, MSCA is equivalent to the Submodular Facility Location problem from [39], where a tight $O(\log n)$ approximation is given. If the functions f_i are nonnegative and nonmonotone, then no multiplicative factor approximation exists [7]. However, the works of [5] and [7] respectively show that $O(\log n)$ and $O(k \log n)$ approximations are available for some special types of nonnegative nonmonotone objectives.

Goel et al [12] consider the minimization case of MASO($\mathcal{F}$) where the objectives are nonnegative and monotone, and $\mathcal{F}$ is a nontrivial collection of subsets of V (i.e. $\mathcal{F} \subset 2^V$). In particular, given a graph G they consider the families of vertex covers, spanning trees, perfect matchings, and shortest *st* paths. They provide a tight $O(\log n)$ approximation for the vertex cover problem, and show polynomial hardness for the other cases. To the best of our knowledge, [12] is the only work on MASO($\mathcal{F}$) for nontrivial collections $\mathcal{F}$.

1.3 Our contributions

We first discuss the minimization side. Here we mainly focus on nonnegative monotone objectives f_i, due to the strong hardness results discussed in Section 1.2. Our main result is showing that if the SA primitive for a family $\mathcal{F}$ admits approximation via a natural "blocking" convex relaxation (see Section 2.1), then we may extend this to its MA version with a modest blow-up in the approximation factor.

▶ **Theorem 6.** *Suppose there is a (polytime) $\alpha(n)$-approximation for monotone SO($\mathcal{F}$) minimization via the blocking convex relaxation. Then there is a (polytime) $O(\alpha(n) \cdot \min\{k, \log^2(n)\})$ approximation for monotone MASO($\mathcal{F}$) minimization.*

We remark that the $O(\log^2(n))$ factor loss due to having multiple agents (i.e the MA gap) is in the right ballpark, since the vertex cover problem has a factor 2-approximation for single-agent and a tight $O(\log n)$-approximation for the MA version [12].

We also discuss how Goel et al's $O(\log n)$-approximation for MA vertex cover is a special case of a more general phenomenon. Their analysis only relies on the fact that the feasible family (or at least its upwards closure) has a *bounded blocker property*. Given a family $\mathcal{F}$, the *blocker* $\mathcal{B}(\mathcal{F})$ of $\mathcal{F}$ consists of the minimal sets B such that $B \cap F \neq \emptyset$ for each $F \in \mathcal{F}$. We say that $\mathcal{B}(\mathcal{F})$ is β-*bounded* if $|B| \leq \beta$ for all $B \in \mathcal{B}(\mathcal{F})$.

Families with bounded blockers have been previously studied in the SA minimization setting, where the works [23, 21] show that β-approximations are always available. Our next result then establishes an $O(\log n)$ MA gap for bounded blocker families, thus improving the $O(\log^2(n))$ factor in Theorem 6 for general families. We remark that this $O(\log n)$ MA gap is tight due to examples like vertex covers [12], or submodular facility location (a trivial 1-approximation for SA and a tight $O(\log n)$-approximation [39] for MA).

▶ **Theorem 7.** *Let $\mathcal{F}$ be a family with a β-bounded blocker. Then there is a randomized $O(\beta \log n)$-approximation algorithm for monotone MASO($\mathcal{F}$) minimization.*

While our work mainly focuses on monotone objectives, in the full version [35] we show that upwards closed families with a bounded blocker remain tractable under some special types of nonmonotone objectives introduced by Chekuri and Ene (see Section 2.3).

We conclude our minimization work by discussing a class of families which behaves well for MA minimization despite not having a bounded blocker. More specifically, we observe in Section 2.4 that crossing (and ring) families have an MA gap of $O(\log n)$.

▶ **Theorem 8.** *There is a tight* $\ln(n)$*-approximation for monotone MASO($\mathcal{F}$) minimization over crossing families $\mathcal{F}$.*

We now discuss our contributions for maximization. Our main result here establishes that if the SA primitive for a family $\mathcal{F}$ admits approximation via its multilinear relaxation (see Section 3.2), then we may extend this to its MA version with a constant factor loss.

▶ **Theorem 9.** *If there is a (polytime) $\alpha(n)$-approximation for monotone SO($\mathcal{F}$) maximization via its multilinear relaxation, then there is a (polytime) $(1-1/e)\cdot\alpha(n)$-approximation for monotone MASO($\mathcal{F}$) maximization. Furthermore, given a downwards closed family $\mathcal{F}$, if there is a (polytime) $\alpha(n)$-approximation for nonmonotone SO($\mathcal{F}$) maximization via its multilinear relaxation, then there is a (polytime) $0.385\cdot\alpha(n)$-approximation for nonmonotone MASO($\mathcal{F}$) maximization.*

We remark that the $(1-1/e)$ MA gap in the monotone case is tight due to examples like $\mathcal{F} = \{V\}$, where there is a trivial 1-approximation for the SA problem and a tight $(1-1/e)$-approximation for the MA version [40].

In Section 3 we describe a simple generic reduction that shows that for some families an (optimal) MA gap of 1 holds.

▶ **Theorem 10.** *Let $\mathcal{F}$ be a matroid, a p-matroid intersection, or a p-system. Then, if there is a (polytime) α-approximation algorithm for monotone (resp. nonmonotone) SO($\mathcal{F}$) maximization, there is a (polytime) α-approximation algorithm for monotone (resp. nonmonotone) MASO($\mathcal{F}$) maximization.*

In the setting of CMASO (i.e. (3)) our results from Section 3.1 provide additional modelling flexibility. They imply that one maintains decent approximations even while adding interesting side constraints. For instance, for a monotone maximization instance of CMASO where $\mathcal{F}$ corresponds to a p-matroid intersection and the $\mathcal{F}_i$ are all matroids, our results lead to a $(p+1+\epsilon)$-approximation algorithm. We believe that these, combined with other results from Section 3, substantially expand the family of tractable models for maximization.

2 Multi-agent submodular minimization

In this section we seek generic reductions for multi-agent minimization problems to their single-agent primitives. We work with a natural blocking convex relaxation that is obtained via the Lovász extension of a set function. We show that if the SA primitive admits approximation via such relaxation, then we may extend this to its MA version up to an $O(\min\{k, \log^2(n)\})$ factor loss. Moreover, as noted already, the $O(\log^2(n))$ approximation factor loss due to having multiple agents is in the right ballpark given the tight $O(\log n)$ MA gap for the vertex cover problem [12]. In Section 2.3 we discuss an extension of this vertex cover result to a larger class of families with a MA gap of $O(\log n)$.

2.1 The single-agent and multi-agent formulations

Due to monotonicity, one may often assume that we are working with a family $\mathcal{F}$ which is *upwards-closed*, aka a *blocking family* (cf. [21]). The advantage is that to certify whether $F \in \mathcal{F}$, we only need to check that $F \cap B \neq \emptyset$ for each element B of the family $\mathcal{B}(\mathcal{F})$ of minimal blockers of $\mathcal{F}$. We discuss the details in Appendix A.

For a set function $f : \{0,1\}^V \to \mathbb{R}$ with $f(\emptyset) = 0$ one defines its *Lovász extension* $f^L : \mathbb{R}^V_+ \to \mathbb{R}$ (introduced in [30]) as follows. Let $0 < v_1 < v_2 < ... < v_m$ be the distinct positive values taken in some vector $z \in \mathbb{R}^V_+$, and let $v_0 = 0$. For each $i \in \{0, 1, ..., m\}$ let $S_i := \{j : z_j > v_i\}$. In particular, S_0 is the support of z and $S_m = \emptyset$. One then defines

$$f^L(z) = \sum_{i=0}^{m-1} (v_{i+1} - v_i) f(S_i).$$

It follows from the definition that f^L is positively homogeneous, that is $f^L(\alpha z) = \alpha f^L(z)$ for any $\alpha > 0$ and $z \in \mathbb{R}^V_+$. It is also straightforward that f^L is monotone if f is. We use both of these properties of f^L in our proofs. We also have the following result due to Lovász.

▶ **Lemma 11** (Lovász [30]). *The function f^L is convex if and only if f is submodular.*

This gives rise to natural convex relaxations for the single-agent and multi-agent problems based on the blocking formulation $P^*(\mathcal{F}) := \{z \geq 0 : z(B) \geq 1 \text{ for all } B \in \mathcal{B}(\mathcal{F})\}$ (see Appendix A for details). The *single-agent Lovász extension formulation* (used in [20, 21]) is:

$$\text{(SA-LE)} \qquad \min f^L(z) : z \in P^*(\mathcal{F}), \tag{4}$$

and the *multi-agent Lovász extension formulation* (used in [5] for $\mathcal{F} = \{V\}$) is:

$$\text{(MA-LE)} \qquad \min \sum_{i \in [k]} f_i^L(z_i) : z_1 + z_2 + \cdots + z_k \in P^*(\mathcal{F}). \tag{5}$$

By standard methods (see Appendix B) one may solve these problems in polytime if one can separate over the blocking formulation $P^*(\mathcal{F})$. This is the case for many natural families such as spanning trees, perfect matchings, *st*-paths, and vertex covers.

It is shown in [5] that in the setting of monotone objectives and $\mathcal{F} = \{V\}$, a fractional solution of (MA-LE) can be rounded into an integral one at an $O(\log n)$ factor loss.

▶ **Theorem 12** ([5]). *Let $z_1 + z_2 + \cdots + z_k$ be a feasible solution for (MA-LE) in the setting where $\mathcal{F} = \{V\}$ (i.e. $\sum_{i \in [k]} z_i = \chi^V$) and f_i are nonnegative monotone submodular. Then there is a randomized rounding procedure that outputs an integral feasible solution $\bar{z}_1 + \bar{z}_2 + \cdots + \bar{z}_k$ such that $\sum_{i \in [k]} f_i^L(\bar{z}_i) \leq O(\log n) \sum_{i \in [k]} f_i^L(z_i)$ on expectation. That is, we get a partition $S_1, S_2, \ldots, S_k$ of V such that $\sum_{i \in [k]} f_i(S_i) \leq O(\log n) \sum_{i \in [k]} f_i^L(z_i)$ on expectation.*

2.2 A multi-agent gap of $O(\min\{k, \log^2(n)\})$

In this section we present the proof of Theorem 6. The high level idea is that we start with an optimal solution $z^* = z_1^* + z_2^* + \cdots + z_k^*$ to the multi-agent relaxation (MA-LE) and build a new feasible solution $\hat{z} = \hat{z}_1 + \hat{z}_2 + \cdots + \hat{z}_k$ where the $\hat{z}_i$ have supports V_i that are pairwise disjoint. We interpret the V_i as the set of items associated (or pre-assigned) to agent i. Once we have such a pre-assignment we consider the single-agent problem $\min g(S) : S \in \mathcal{F}$ where

$$g(S) = \sum_{i \in [k]} f_i(S \cap V_i). \tag{6}$$

It is clear that g is nonnegative monotone submodular since the f_i are as well. Moreover, for any solution $S \in \mathcal{F}$ for this single-agent problem we obtain a MA solution of the same cost by setting $S_i = S \cap V_i$, since we then have $g(S) = \sum_{i \in [k]} f_i(S \cap V_i) = \sum_{i \in [k]} f_i(S_i)$.

For a set $S \subseteq V$ and a vector $z \in [0,1]^V$ we denote by $z|_S$ the truncation of z to elements of S. That is, we set $z|_S(v) = z(v)$ for each $v \in S$ and to zero otherwise. We then have by definition of g that $g^L(z) = \sum_{i \in [k]} f_i^L(z|_{V_i})$. Moreover, if the V_i are pairwise disjoint, then we also have $\sum_{i \in [k]} f_i^L(z|_{V_i}) = \sum_{i \in [k]} f_i^L(z_i)$. We summarize this in the following result.

▶ **Proposition 13.** *Let $z = z_1 + z_2 + \cdots + z_k$ be a feasible solution to (MA-LE) where the vectors z_i have pairwise disjoint supports V_i. Then $g^L(z) = \sum_{i \in [k]} f_i^L(z|_{V_i}) = \sum_{i \in [k]} f_i^L(z_i)$.*

The next two results show how one can get a feasible solution $\hat{z} = \hat{z}_1 + \hat{z}_2 + \cdots + \hat{z}_k$ where the $\hat{z}_i$ have pairwise disjoint supports, by losing a factor of $O(\log^2(n))$ and k respectively. We remark that these two results combined prove Theorem 6.

▶ **Theorem 14.** *Suppose there is a (polytime) $\alpha(n)$-approximation for monotone SO($\mathcal{F}$) minimization based on rounding (SA-LE). Then there is a (polytime) $O(\alpha(n)\log^2(n))$-approximation for monotone MASO($\mathcal{F}$) minimization.*

Proof. Let $z^* = z_1^* + z_2^* + \cdots + z_k^*$ denote an optimal solution to (MA-LE) with value OPT_{frac}. In order to apply a black box single-agent rounding algorithm we must create a different multi-agent solution. This is done in several steps, the first few of which are standard. The key steps are *fracture, expand and return*, which arise later in the process.

Call an element v *small* if $z^*(v) \leq \frac{1}{2n}$. Note that $\sum_{v \text{ small}} z^*(v) \leq \frac{1}{2}$, and so for any blocking set $B \in \mathcal{B}(\mathcal{F})$, we have that at most $\frac{1}{2}$ of $z^*(B)$ is contributed by small elements. We obtain a new feasible solution $z' = z_1' + z_2' + \cdots + z_k'$ by removing all small elements from the support of the z_i^* and then doubling the resulting vectors. By the monotonicity and homogeneity of the f_i^L, this at most doubles the cost of OPT_{frac}.

We now prune the solution $z' = z_1' + z_2' + \cdots + z_k'$ a bit more. Let Z_j be the elements v such that $z'(v) \in (2^{-(j+1)}, 2^{-j}]$ for $j = 0, 1, 2, \ldots, L$. Since $z'(v) > \frac{1}{2n}$ for any element in the support, we have that $L \leq \log(n)$. We call Z_j *bin* j and define $r_j = 2^j$. We round up each $v \in Z_j$ so that $z'(v) = 2^{-j}$ by augmenting the z_i' values by at most a factor of 2. We may do this simultaneously for all v by possibly "truncating" the values associated to some of the elements. As before, this is fine since the f_i^L are monotone. In the end, we call this a *uniform solution* $z'' = z_1'' + z_2'' + \cdots + z_k''$ in the sense that each $z''(v)$ is some power of 2. Note that its cost is at most $4 \cdot OPT_{frac}$.

Fracture. We now *fracture* the vectors z_i'' by defining vectors $z_{i,j}'' = z_i''|_{Z_j}$ for each $i \in [k]$ and each $j \in \{0, 1, \ldots, L\}$, where recall that the notation $z|_S$ denotes the truncation of z to elements of S. Notice that $z_i'' = \sum_{j=0}^{L} z_{i,j}''$.

Expand. Now for each $j \in \{0, 1, \ldots, L\}$ we blow up the vectors $z_{i,j}''$ by a factor r_j. Since $z''(v) = \frac{1}{r_j}$ for each $v \in Z_j$, the resulting values yield a (probably fractional) cover of Z_j. We

can then use the rounding procedure discussed in Theorem 12 (with ground set Z_j) to get an integral solution $z'''_{i,j}$ such that $\sum_i f_i^L(z'''_{i,j}) \leq O(\log n) \sum_i f_i^L(r_j \cdot z''_{i,j})$ on expectation.

Return. Now we go back to get a new MA-LE solution $\hat{z} = \hat{z}_1 + \hat{z}_2 + \cdots + \hat{z}_k$ by setting $\hat{z}_i = \sum_{j=0}^{L} \frac{1}{r_j} z'''_{i,j}$. Note that $\hat{z} = z''$ and so this is indeed feasible (and again uniform). Moreover, we have that the cost of this new solution satisfies

$$\begin{aligned}
\sum_{i=1}^{k} f_i^L(\hat{z}_i) &\leq \sum_{i=1}^{k}\sum_{j=0}^{L} \frac{1}{r_j} f_i^L(z'''_{i,j}) = \sum_{j=0}^{L} \frac{1}{r_j} \sum_{i=1}^{k} f_i^L(z'''_{i,j}) \leq O(\log n) \sum_{i=1}^{k}\sum_{j=0}^{L} f_i^L(z''_{i,j}) \\
&\leq O(\log n)(L+1) \sum_{i=1}^{k} f_i^L(z''_i) \\
&\leq O(\log^2(n)) \sum_{i=1}^{k} f_i^L(z^*_i) \leq O(\log^2(n)) \cdot OPT_{MA},
\end{aligned}$$

where in the first inequality we use the convexity and homogeneity of the f_i^L, in the second inequality we use again the homogeneity together with the upper bound for $\sum_i f_i^L(z'''_{i,j})$, and in the third inequality we use monotonicity and the fact that $z''_{i,j} \leq z''_i$ for all j.

Single-Agent Rounding. In the last step we use the function g defined in (6), with sets V_i corresponding to the support of the $\hat{z}_i$. Given our α-approximation rounding assumption for (SA-LE), we can round $\hat{z}$ to find a set $\hat{S}$ such that $g(\hat{S}) \leq \alpha g^L(\hat{z})$. Then, by setting $\hat{S}_i = \hat{S} \cap V_i$ we obtain a MA solution satisfying

$$\sum_{i=1}^{k} f_i(\hat{S}_i) = g(\hat{S}) \leq \alpha g^L(\hat{z}) = \alpha \sum_{i=1}^{k} f_i^L(\hat{z}_i) \leq \alpha \cdot O(\log^2(n)) \cdot OPT_{MA},$$

where the second equality follows from Proposition 13. This completes the proof. ◀

While in this paper we define our (SA-LE) and (MA-LE) formulations in terms of the blocking formulation $P^*(\mathcal{F})$, in the full version [35] we discuss how our results naturally extend to more general upwards closed relaxations $\{z \geq 0 : Az \geq r\}$ of the integral polyhedron $conv(\{\chi^S : S \in \mathcal{F}\})$. We also show how a more careful analysis of the above proof leads to a slightly improved MA gap of $O(\log(n) \log(\frac{n}{\log n}))$.

We now give an approximation in terms of the number of agents, which becomes preferable in settings where $k < \log^2(n)$.

▶ **Lemma 15.** *Suppose there is a (polytime) $\alpha(n)$-approximation for monotone SO($\mathcal{F}$) minimization based on rounding (SA-LE). Then there is a (polytime) $k\alpha(n)$-approximation for monotone MASO($\mathcal{F}$) minimization.*

Proof. Let $z^* = z_1^* + z_2^* + \cdots + z_k^*$ denote an optimal solution to (MA-LE) with value OPT_{frac}. We build a new feasible solution $\hat{z} = \hat{z}_1 + \hat{z}_2 + \cdots + \hat{z}_k$ as follows. For each element $v \in V$ let $i' = \text{argmax}_{i \in [k]} z_i^*(v)$, breaking ties arbitrarily. Then set $\hat{z}_{i'}(v) = k z_i^*(v)$ and $\hat{z}_i(v) = 0$ for each $i \neq i'$. By construction we have $\hat{z} \geq z^*$, and hence this is indeed a feasible solution. Moreover, by construction we also have that $\hat{z}_i \leq k z_i^*$ for each $i \in [k]$. Hence, given the monotonicity and homogeneity of the f_i^L we have

$$\sum_{i \in [k]} f_i^L(\hat{z}_i) \leq \sum_{i \in [k]} f_i^L(k z_i^*) = k \sum_{i \in [k]} f_i^L(z_i^*) = k \cdot OPT_{frac} \leq k \cdot OPT_{MA}.$$

Since the $\hat{z}_i$ have disjoint supports V_i, we can now use the function g defined in (6) and do a single-rounding argument as in Theorem 14. This completes the proof. ◀

2.3 A tight multi-agent gap of $O(\log n)$ for bounded blocker families

While we established an $O(\log^2(n))$ MA gap for general families based on the blocking convex formulations, the work of [12] shows an improved MA gap of $O(\log n)$ for vertex covers. In this section we generalize their result by describing a larger family class with such MA gap.

Their algorithm relies on the fact that the set family has the following *bounded blocker property*. We call a clutter (family of non-comparable sets) $\mathcal{F}$ *β-bounded* if $|F| \leq \beta$ for all $F \in \mathcal{F}$. Recall that the *blocker* of a clutter $\mathcal{F}$, denoted by $\mathcal{B}(\mathcal{F})$, is the set of all minimal B such that $B \cap F \neq \emptyset$ for all $F \in \mathcal{F}$. We then say that $\mathcal{F}$ has a β-bounded blocker if $|B| \leq \beta$ for each $B \in \mathcal{B}(\mathcal{F})$. The main SA minimization result for such families is the following.

▶ **Theorem 16** ([21, 23]). *Let $\mathcal{F}$ be a family with a β-bounded blocker. Then there is a β-approximation algorithm for monotone $SO(\mathcal{F})$ minimization. If $P^*(\mathcal{F})$ has a polytime separation oracle, then this is a polytime algorithm.*

Our next result establishes an $O(\log n)$ MA gap for families with a bounded blocker.

▶ **Theorem 17.** *Let $\mathcal{F}$ be an upwards closed family with a β-bounded blocker. Then there is a randomized $O(\beta \log n)$-approximation algorithm for the monotone $MASO(\mathcal{F})$ minimization problem. If $P^*(\mathcal{F})$ has a polytime separation oracle, then this is a polytime algorithm.*

Proof. Let $z^* = \sum_{i\in[k]} z_i^*$ be an optimal solution to (MA-LE) with value OPT_{frac}. Consider the new feasible solution given by $\beta z^* = \sum_{i\in[k]} \beta z_i^*$ and let $U = \{v \in V : \beta z^*(v) \geq 1\}$. Since $\mathcal{F}$ has a β-bounded blocker it follows that $U \in \mathcal{F}$. We now have that $\sum_{i\in[k]} \beta z_i^*$ is a feasible solution such that $\sum_{i\in[k]} \beta z_i^* \geq \chi^U$. Thus, we can use Theorem 12 to get an integral feasible solution $\sum_{i\in[k]} \bar{z}_i$ such that $\sum_{i\in[k]} \bar{z}_i \geq \chi^U$ and $\sum_{i\in[k]} f_i^L(\bar{z}_i) \leq O(\log |U|) \sum_{i\in[k]} f_i^L(\beta z_i^*) \leq \beta \cdot O(\log n) \cdot OPT_{frac}$ on expectation. ◀

While our work focuses on monotone objectives, in the full version [35] we show that upwards closed families with a bounded blocker remain tractable under some special types of nonmonotone objectives. These were introduced in [5] and [7], where they consider objectives of the form $f_i = g_i + h$ where the g_i are monotone submodular and h is symmetric submodular (in [5]) or just submodular (in [7]). Note that by taking $h \equiv 0$ (which is symmetric submodular) we recover the monotone case.

2.4 A tight multi-agent gap of $O(\log n)$ for ring and crossing families

It is well known ([36]) that submodular minimization can be solved exactly in polynomial time over a ring family. In this section we observe that the MA problem over this type of constraint admits a tight $\ln(n)$-approximation. More generally, we consider *crossing families*. A family $\mathcal{F}$ of subsets of V forms a ring family (aka lattice family) if for each $A, B \in \mathcal{F}$ we have $A \cap B, A \cup B \in \mathcal{F}$. A crossing family is one where we only require it for sets where $A \setminus B, B \setminus A, A \cap B, V - (A \cup B)$ are all non-empty. Thus any ring family is a crossing family.

For any crossing family $\mathcal{F}$ and any $u, v \in V$, let $\mathcal{F}_{uv} = \{A \in \mathcal{F} : u \in A, v \notin A\}$. It is easy to see that $\mathcal{F}_{uv}$ is a ring family. We may then solve the original MA problem by solving the associated MA problem for each non-empty $\mathcal{F}_{uv}$ and then selecting the best output solution.

So we assume now that we are given a ring family in such a way that we may compute its minimal set M (which is unique). This is a standard assumption when working with ring families (cf. submodular minimization algorithm described in [36]). Then, due to monotonicity and the fact that $\mathcal{F}$ is closed under intersections, it is not hard to see that the

original problem reduces to the submodular facility location problem

$$\min \sum_{i=1}^{k} f_i(S_i) : S_1 \uplus \cdots \uplus S_k = M \ ,$$

which admits a tight $(\ln |M|)$-approximation ([39]). In particular, for the special case where we have the trivial ring family $\mathcal{F} = \{V\}$ we get a tight $\ln(n)$-approximation. The next result summarizes these observations.

▶ **Theorem 18.** *There is a tight* $\ln(n)$*-approximation for monotone* $MASO(\mathcal{F})$ *minimization over crossing families* $\mathcal{F}$*.*

3 Multi-agent submodular maximization

In this section we describe two different reductions. The first one reduces the capacitated multi-agent problem (3) to a single-agent problem. We show that several properties of the objective and family of feasible sets stay *invariant* (i.e. preserved) under the reduction. We use this to establish an (optimal) MA gap of 1 for several families. Examples of such families include spanning trees, matroids, and p-systems.

Our second reduction uses the multilinear extension of a set function. We establish that if the SA (monotone or nonmonotone) primitive admits approximation via its multilinear relaxation, then we may extend this to its MA version with a constant factor loss. Moreover, for the monotone case our MA gap is tight.

3.1 The lifting reduction

In this section we describe a generic reduction of CMASO (i.e. (3)) to a single-agent problem: $\max / \min f(S) : S \in \mathcal{L}$. The argument is based on the idea of viewing assignments of elements v to agents i in a *multi-agent bipartite graph*. This simple idea (which is equivalent to making k disjoint copies of the ground set) already appeared in the classical work of Fisher et al [10], and has since then been widely used [29, 40, 4, 37]. We review briefly the reduction here for completeness and to fix notation.

Consider the complete bipartite graph $G = ([k] + V, E)$. Every subset of edges $S \subseteq E$ can be written uniquely as $S = \uplus_{i\in[k]}(\{i\} \times S_i)$ for some sets $S_i \subseteq V$. This allows us to go from a multi-agent objective (such as the one in (3)) to a univariate objective $f : 2^E \to \mathbb{R}$ over the lifted space. Namely, for each set $S \subseteq E$ we define $f(S) = \sum_{i\in[k]} f_i(S_i)$. The function f is well-defined because each subset $S \subseteq E$ can be uniquely written as $S = \uplus_{i\in[k]}(\{i\} \times S_i)$.

We consider two families of sets over E that capture the original constraints:

$$\mathcal{F}' := \{S \subseteq E : S_1 \uplus \cdots \uplus S_k \in \mathcal{F}\} \quad \text{and} \quad \mathcal{H} := \{S \subseteq E : S_i \in \mathcal{F}_i,\ \forall i \in [k]\}.$$

We now have:

$$\begin{array}{ccc} \max / \min & \sum_{i\in[k]} f_i(S_i) & \\ \text{s.t.} & S_1 \uplus \cdots \uplus S_k \in \mathcal{F} & \\ & S_i \in \mathcal{F}_i \ , \forall i \in [k] & \end{array} = \begin{array}{ccc} \max / \min & f(S) & \\ \text{s.t.} & S \in \mathcal{F}' \cap \mathcal{H} & \end{array} = \begin{array}{ccc} \max / \min & f(S) & \\ \text{s.t.} & S \in \mathcal{L} & \ , \end{array}$$

where in the last step we just let $\mathcal{L} := \mathcal{F}' \cap \mathcal{H}$.

Clearly, this reduction is interesting if our new function f and family of sets $\mathcal{L}$ have properties which allows us to handle them computationally. This will depend on the original structure of the functions f_i, and the set families $\mathcal{F}$ and $\mathcal{F}_i$. In terms of the objective, it is

Table 1 Invariant properties under the lifting reduction.

	Multi-agent problem	**Single-agent (i.e. reduced) problem**	Result
1	$(V, \mathcal{F})$ a p-system	$(E, \mathcal{F}')$ a p-system	Appendix C
2	$\mathcal{F}$ = bases of a p-system	$\mathcal{F}'$ = bases of a p-system	Appendix C
3	$(V, \mathcal{F})$ a matroid	$(E, \mathcal{F}')$ a matroid	Appendix C
4	$\mathcal{F}$ = bases of a matroid	$\mathcal{F}'$ = bases of a matroid	Appendix C
5	$(V, \mathcal{F})$ a p-matroid intersection	$(E, \mathcal{F}')$ a p-matroid intersection	Full version [35]
6	$\mathcal{F}$ = forests (resp. spanning trees)	$\mathcal{F}'$ = forests (resp. spanning trees)	Full version [35]
7	$\mathcal{F}$ = matchings (resp. perfect matchings)	$\mathcal{F}'$ = matchings (resp. perfect matchings)	Full version [35]
8	$\mathcal{F}$ = st-paths	$\mathcal{F}'$ = st-paths	Full version [35]
9	$(V, \mathcal{F}_i)$ a matroid for all $i \in [k]$	$(E, \mathcal{H})$ a matroid	Full version [35]
10	$\mathcal{F}_i$ a ring family for all $i \in [k]$	$\mathcal{H}$ a ring family	Full version [35]

straightforward to check (as previously pointed out in [29]) that if the f_i are (nonnegative, respectively monotone) submodular functions, then f as defined above is also (nonnegative, respectively monotone) submodular. In the full version [35] we discuss several properties of the families $\mathcal{F}$ and $\mathcal{F}_i$ that are preserved under this reduction, as well as their algorithmic consequences. We show, for instance, that if the family $\mathcal{F}$ induces a matroid (or more generally a p-system) over the original ground set V, then so does the family $\mathcal{F}'$ over the lifted space E. We summarize some of these results in Table 1, and we remark that these now prove Theorem 10 (see [35] for full details).

3.2 The single-agent and multi-agent formulations

For a set function $f : \{0,1\}^V \to \mathbb{R}$ we define its *multilinear extension* $f^M : [0,1]^V \to \mathbb{R}$ (introduced in [3]) as

$$f^M(z) = \sum_{S \subseteq V} f(S) \prod_{v \in S} z_v \prod_{v \notin S} (1 - z_v).$$

An alternative way to define f^M is in terms of expectations. Consider a vector $z \in [0,1]^V$ and let R^z denote a random set that contains element v_i independently with probability z_{v_i}. Then $f^M(z) = \mathbb{E}[f(R^z)]$, where the expectation is taken over random sets generated from the probability distribution induced by z.

This gives rise to natural single-agent and multi-agent relaxations for constrained submodular maximization. The *single-agent multilinear extension relaxation* is:

$$\text{(SA-ME)} \qquad \max f^M(z) : z \in P(\mathcal{F}), \tag{7}$$

and the *multi-agent multilinear extension relaxation* is:

$$\text{(MA-ME)} \qquad \max \sum_{i \in [k]} f_i^M(z_i) : z_1 + z_2 + \cdots + z_k \in P(\mathcal{F}), \tag{8}$$

where $P(\mathcal{F})$ denotes some fractional relaxation of the integral polytope $conv(\{\chi^S : S \in \mathcal{F}\})$. While the relaxation (SA-ME) has been used extensively [4, 27, 9, 6, 1] in the submodular maximization literature, we are not aware of any previous work using the multi-agent relaxation (MA-ME). We next discuss the solvability of (SA-ME).

▶ **Theorem 19** ([1, 40]). *Let $f : 2^V \to \mathbb{R}_+$ be nonnegative submodular and f^M its multilinear extension. Let $P \subseteq [0,1]^V$ be any downwards closed polytope that admits a polytime separation*

oracle, and let $OPT = \max f^M(z) : z \in P$. *Then there is a polytime algorithm ([1]) that finds* $z^* \in P$ *such that* $f^M(z^*) \geq 0.385 \cdot OPT$. *Moreover, if* f *is monotone there is a polytime algorithm ([40]) that finds* $z^* \in P$ *such that* $f^M(z^*) \geq (1 - 1/e)OPT$.

For monotone objectives the assumption that P is downwards closed is without loss of generality. This is not the case, however, when the objective is nonmonotone. Nonetheless, this restriction is unavoidable, as Vondrák [41] showed that no algorithm can find $z^* \in P$ such that $f^M(z^*) \geq c \cdot OPT$ for any constant $c > 0$ when P admits a polytime separation oracle but it is not downwards closed.

We remark that we can solve (MA-ME) to the same approximation factor as (SA-ME). This follows from the fact that the MA problem has the form $\{\max g(w) : w \in W \subseteq \mathbf{R}^{nk}\}$ where $g(w) = g(z_1, z_2, \ldots, z_k) = \sum_{i\in[k]} f_i^M(z_i)$ and W is the downwards closed polytope $\{w = (z_1, ..., z_k) : \sum_i z_i \in P(\mathcal{F})\}$. Clearly we have a polytime separation oracle for W given that we have one for $P(\mathcal{F})$. Moreover, it is straightforward to check (see Lemma 34 on Appendix C) that $g(w) = f^M(w)$, where f is the function on the lifted space after applying the lifting reduction from Section 3.1. Thus, g is the multilinear extension of a nonnegative submodular function, and we can use Theorem 19.

3.3 A tight multi-agent gap of $1 - 1/e$

In this section we present the proof of Theorem 9. The high-level idea behind our reduction is the same as in the minimization setting (see Section 2.2). That is, we start with an (approximate) optimal solution $z^* = z_1^* + z_2^* + \cdots + z_k^*$ to the multi-agent (MA-ME) relaxation and build a new feasible solution $\hat{z} = \hat{z}_1 + \hat{z}_2 + \cdots + \hat{z}_k$ where the $\hat{z}_i$ have supports V_i that are pairwise disjoint. We then use for the SA rounding step the single-agent problem (as previously defined in (6) for the minimization setting) $\max g(S) : S \in \mathcal{F}$ where $g(S) = \sum_{i\in[k]} f_i(S \cap V_i)$.

Similarly to Proposition 13 which dealt with the Lovász extension, we have the following result for the multilinear extension.

▶ **Proposition 20.** *Let* $z = \sum_{i\in[k]} z_i$ *be a feasible solution to (MA-ME) where the vectors* z_i *have pairwise disjoint supports* V_i. *Then* $g^M(z) = \sum_{i\in[k]} f_i^M(z|_{V_i}) = \sum_{i\in[k]} f_i^M(z_i)$.

We now have all the ingredients to prove our main result in the maximization setting.

▶ **Theorem 21.** *If there is a (polytime)* $\alpha(n)$*-approximation for monotone SO(*$\mathcal{F}$*) maximization via rounding (SA-ME), there is a (polytime)* $(1-1/e)\cdot\alpha(n)$*-approximation for monotone MASO(*$\mathcal{F}$*) maximization. Furthermore, if* $\mathcal{F}$ *is downwards closed and there is a (polytime)* $\alpha(n)$*-approximation for nonmonotone SO(*$\mathcal{F}$*) maximization via rounding (SA-ME), there is a (polytime)* $0.385 \cdot \alpha(n)$*-approximation for nonmonotone MASO(*$\mathcal{F}$*) maximization.*

Proof. We discuss first the case of monotone objectives.

STEP 1. Let $z^* = z_1^* + z_2^* + \cdots + z_k^*$ denote an approximate solution to (MA-ME) obtained via Theorem 19, and let OPT_{frac} be the value of an optimal solution. We then have that $\sum_{i\in[k]} f_i^M(z_i^*) \geq (1 - 1/e)OPT_{frac} \geq (1 - 1/e)OPT_{MA}$.

STEP 2. For an element $v \in V$ let $\mathbf{e_v}$ denote the characteristic vector of $\{v\}$, i.e. the vector in $\mathbb{R}^V$ which has value 1 in the v-th component and zero elsewhere. Notice that by definition of the multilinear extension we have that the functions f_i^M are linear along directions $\mathbf{e_v}$ for any $v \in V$. It then follows that the function

$$h(t) = f_i^M(z_i^* + t\mathbf{e_v}) + f_{i'}^M(z_{i'}^* - t\mathbf{e_v}) + \sum_{j\in[k],j\neq i,i'} f_j^M(z_j^*)$$

is also linear for any $v \in V$ and $i \neq i' \in [k]$, since it is the sum of linear functions (on t). In particular, given any $v \in V$ such that there exist $i \neq i' \in [k]$ with $z_i^*(v), z_{i'}^*(v) > 0$, there is always a choice so that increasing one component and decreasing the other by the same amount does not decrease the objective value. We use this as follows.

Let $v \in V$ be such that there exist $i \neq i' \in [k]$ with $z_i^*(v), z_{i'}^*(v) > 0$. Then, we either set $z_i^*(v) = z_i^*(v) + z_{i'}^*(v)$ and $z_{i'}^*(v) = 0$, or $z_{i'}^*(v) = z_i^*(v) + z_{i'}^*(v)$ and $z_i^*(v) = 0$, whichever does not decrease the objective value. We repeat until the vectors z_i^* have pairwise disjoint support. Let us denote these new vectors by $\hat{z}_i$ and let $\hat{z} = \sum_{i\in[k]} \hat{z}_i$. Then notice that the vector $z^* = \sum_{i\in[k]} z_i^*$ remains invariant after performing each of the above updates (i.e. $\hat{z} = z^*$), and hence the new vectors $\hat{z}_i$ remain feasible.

STEP 3. In the last step we use the function g defined in (6), with sets V_i corresponding to the support of the $\hat{z}_i$. Given our α-approximation rounding assumption for (SA-ME), we can round $\hat{z}$ to find a set $\hat{S}$ such that $g(\hat{S}) \geq \alpha g^M(\hat{z})$. Then, by setting $\hat{S}_i = \hat{S} \cap V_i$ we obtain a MA solution satisfying

$$\sum_{i=1}^{k} f_i(\hat{S}_i) = g(\hat{S}) \geq \alpha g^M(\hat{z}) = \alpha \sum_{i=1}^{k} f_i^M(\hat{z}_i) \geq \alpha \sum_{i=1}^{k} f_i^M(z_i^*) \geq \alpha(1-1/e)OPT_{MA},$$

where the second equality uses Proposition 20. This completes the monotone case.

For the nonmonotone setting the proof is very similar. Here we restrict our attention to downwards closed families, since then we can get a 0.385-approximation at STEP 1 via Theorem 19. We then apply STEP 2 and 3 in the same fashion as we did for monotone objectives. This leads to a $0.385 \cdot \alpha(n)$-approximation for the multi-agent problem. ◀

4 Conclusion

A number of interesting questions remain. Perhaps the main one being whether the $O(\log^2(n))$ MA gap for minimization can be improved to $O(\log n)$? We have shown this is the case for bounded blocker and crossing families. Another question is whether the $\alpha \log^2(n)$ and αk approximations can be made truly black box? I.e., do not depend on the convex formulation.

On separate work ([34]) we discuss multivariate submodular objectives. We show that our reductions for maximization remain well-behaved algorithmically and this opens up more tractable models. This is the topic of planned future work.

References

1. Niv Buchbinder and Moran Feldman. Constrained submodular maximization via a non-symmetric technique. *arXiv preprint arXiv:1611.03253*, 2016.
2. Niv Buchbinder, Moran Feldman, Joseph Seffi, and Roy Schwartz. A tight linear time (1/2)-approximation for unconstrained submodular maximization. *SIAM Journal on Computing*, 44(5):1384–1402, 2015.
3. Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a submodular set function subject to a matroid constraint. In *Integer programming and combinatorial optimization*, pages 182–196. Springer, 2007.
4. Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*, 40(6):1740–1766, 2011.
5. Chandra Chekuri and Alina Ene. Submodular cost allocation problem and applications. In *International Colloquium on Automata, Languages, and Programming*, pages 354–366. Springer, 2011. Extended version: arXiv preprint arXiv:1105.2040.

6 Alina Ene and Huy L Nguyen. Constrained submodular maximization: Beyond 1/e. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 248–257. IEEE, 2016.

7 Alina Ene and Jan Vondrák. Hardness of submodular cost allocation: Lattice matching and a simplex coloring conjecture. *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2014)*, 28:144–159, 2014.

8 Uriel Feige, Vahab S Mirrokni, and Jan Vondrak. Maximizing non-monotone submodular functions. *SIAM Journal on Computing*, 40(4):1133–1153, 2011.

9 Moran Feldman, Joseph Naor, and Roy Schwartz. A unified continuous greedy algorithm for submodular maximization. In *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*, pages 570–579. IEEE, 2011.

10 Marshall L Fisher, George L Nemhauser, and Laurence A Wolsey. *An analysis of approximations for maximizing submodular set functions-II*. Springer, 1978.

11 Lisa Fleischer, Michel X Goemans, Vahab S Mirrokni, and Maxim Sviridenko. Tight approximation algorithms for maximum general assignment problems. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 611–620. Society for Industrial and Applied Mathematics, 2006.

12 Gagan Goel, Chinmay Karande, Pushkar Tripathi, and Lei Wang. Approximability of combinatorial problems with multi-agent submodular cost functions. In *Foundations of Computer Science, 2009. FOCS'09. 50th Annual IEEE Symposium on*, pages 755–764. IEEE, 2009.

13 Michel X. Goemans and VS Ramakrishnan. Minimizing submodular functions over families of sets. *Combinatorica*, 15(4):499–513, 1995.

14 Pranava R Goundan and Andreas S Schulz. Revisiting the greedy approach to submodular set function maximization. *Optimization online*, pages 1–25, 2007.

15 Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2. Springer Science & Business Media, 2012.

16 Anupam Gupta, Aaron Roth, Grant Schoenebeck, and Kunal Talwar. Constrained non-monotone submodular maximization: Offline and secretary algorithms. In *International Workshop on Internet and Network Economics*, pages 246–257. Springer, 2010.

17 Ara Hayrapetyan, Chaitanya Swamy, and Éva Tardos. Network design for information networks. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 933–942. Society for Industrial and Applied Mathematics, 2005.

18 Elad Hazan, Shmuel Safra, and Oded Schwartz. On the complexity of approximating k-set packing. *computational complexity*, 15(1):20–39, 2006.

19 Satoru Iwata, Lisa Fleischer, and Satoru Fujishige. A combinatorial strongly polynomial algorithm for minimizing submodular functions. *Journal of the ACM (JACM)*, 48(4):761–777, 2001.

20 Satoru Iwata and Kiyohito Nagano. Submodular function minimization under covering constraints. In *Foundations of Computer Science, 2009. FOCS'09. 50th Annual IEEE Symposium on*, pages 671–680. IEEE, 2009.

21 Rishabh Iyer, Stefanie Jegelka, and Jeff Bilmes. Monotone closure of relaxed constraints in submodular optimization: Connections between minimization and maximization: Extended version, 2014.

22 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.

23 Christos Koufogiannakis and Neal E Young. Greedy δ-approximation algorithm for covering with arbitrary constraints and submodular cost. *Algorithmica*, 66(1):113–152, 2013.

24 Andreas Krause and Carlos Guestrin. Near-optimal observation selection using submodular functions. In *AAAI*, volume 7, pages 1650–1654, 2007.

25 Andreas Krause, Jure Leskovec, Carlos Guestrin, Jeanne VanBriesen, and Christos Faloutsos. Efficient sensor placement optimization for securing large water distribution networks. *Journal of Water Resources Planning and Management*, 134(6):516–526, November 2008.

26 KW Krause, MA Goodwin, and RW Smith. *Optimal software test planning through automated network analysis*. TRW Systems Group, 1973.

27 Jon Lee, Vahab S Mirrokni, Viswanath Nagarajan, and Maxim Sviridenko. Non-monotone submodular maximization under matroid and knapsack constraints. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 323–332. ACM, 2009.

28 Jon Lee, Maxim Sviridenko, and Jan Vondrák. Submodular maximization over multiple matroids via generalized exchange properties. *Mathematics of Operations Research*, 35(4):795–806, 2010.

29 Benny Lehmann, Daniel Lehmann, and Noam Nisan. Combinatorial auctions with decreasing marginal utilities. *Games and Economic Behavior*, 55(2):270–296, 2006. URL: `http://EconPapers.repec.org/RePEc:eee:gamebe:v:55:y:2006:i:2:p:270-296`.

30 László Lovász. Submodular functions and convexity. In *Mathematical Programming The State of the Art*, pages 235–257. Springer, 1983.

31 Vahab Mirrokni, Michael Schapira, and Jan Vondrák. Tight information-theoretic lower bounds for welfare maximization in combinatorial auctions. In *Proceedings of the 9th ACM conference on Electronic commerce*, pages 70–77. ACM, 2008.

32 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.

33 George L Nemhauser and Leonard A Wolsey. Best algorithms for approximating the maximum of a submodular set function. *Mathematics of operations research*, 3(3):177–188, 1978.

34 Richard Santiago and F Bruce Shepherd. Multivariate submodular optimization. *arXiv preprint arXiv:1612.05222*, 2016.

35 Richard Santiago and F Bruce Shepherd. Multi-agent submodular optimization. *arXiv preprint arXiv:1803.03767*, 2018.

36 Alexander Schrijver. A combinatorial algorithm minimizing submodular functions in strongly polynomial time. *Journal of Combinatorial Theory, Series B*, 80(2):346–355, 2000.

37 Ajit Singh, Andrew Guillory, and Jeff Bilmes. On bisubmodular maximization. In *Artificial Intelligence and Statistics*, pages 1055–1063, 2012.

38 Zoya Svitkina and Lisa Fleischer. Submodular approximation: Sampling-based algorithms and lower bounds. *SIAM Journal on Computing*, 40(6):1715–1737, 2011.

39 Zoya Svitkina and ÉVA Tardos. Facility location with hierarchical facility costs. *ACM Transactions on Algorithms (TALG)*, 6(2):37, 2010.

40 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.

41 Jan Vondrák. Symmetry and approximability of submodular maximization problems. *SIAM Journal on Computing*, 42(1):265–304, 2013.

42 Jan Vondrák, Chandra Chekuri, and Rico Zenklusen. Submodular function maximization via the multilinear relaxation and contention resolution schemes. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 783–792. ACM, 2011.

A Upwards-closed (aka blocking) families

In this section, we give some background for blocking families. As our work for minimization is restricted to monotone functions, we can often convert an arbitrary set family into its upwards-closure (i.e., a blocking version of it) and work with it instead. We discuss this reduction as well. The technical details discussed in this section are fairly standard and we include them for completeness. Several of these results have already appeared in [21].

A.1 Blocking families and a natural relaxation for $P(\mathcal{F})$

A set family $\mathcal{F}$, over a ground set V is *upwards-closed* if $F \subseteq F'$ and $F \in \mathcal{F}$, implies that $F' \in \mathcal{F}$; these are sometimes referred to as *blocking families*. Examples of such families include vertex-covers or set covers more generally, whereas spanning trees are not.

For a blocking family $\mathcal{F}$ one often works with the induced sub-family $\mathcal{F}^{min}$ of minimal sets. Then $\mathcal{F}^{min}$ has the property that it is a *clutter*, that is, $\mathcal{F}^{min}$ does not contain a pair of comparable sets, i.e., sets $F \subset F'$. If $\mathcal{F}$ is a clutter, then $\mathcal{F} = \mathcal{F}^{min}$ and there is an associated *blocking* clutter $\mathcal{B}(\mathcal{F})$, which consists of the minimal sets B such that $B \cap F \neq \emptyset$ for each $F \in \mathcal{F}$. We refer to $\mathcal{B}(\mathcal{F})$ as the *blocker* of $\mathcal{F}$.

One also checks that for an arbitrary upwards-closed family $\mathcal{F}$, we have the following.

▶ **Claim 22** (Lehman)**.**
1. *$F \in \mathcal{F}$ if and only if $F \cap B \neq \emptyset$ for all $B \in \mathcal{B}(\mathcal{F}^{min})$.*
2. *$\mathcal{B}(\mathcal{B}(\mathcal{F}^{min})) = \mathcal{F}^{min}$.*

Thus the significance of blockers is that one may assert membership in an upwards-closed family $\mathcal{F}$ by checking intersections on sets from the blocker $\mathcal{B}(\mathcal{F}^{min})$. If we define $\mathcal{B}(\mathcal{F})$ to be the minimal sets which intersect every element of $\mathcal{F}$, then one checks that $\mathcal{B}(\mathcal{F}) = \mathcal{B}(\mathcal{F}^{min})$. These observations lead to a natural relaxation for minimization problems over the integral polyhedron $P(\mathcal{F}) = conv(\{\chi^F : F \in \mathcal{F}\})$. The *blocking formulation* for $\mathcal{F}$ is:

$$P^*(\mathcal{F}) = \{z \in \mathbb{R}^V_{\geq 0} : z(B) \geq 1 \ \ \forall B \in \mathcal{B}(\mathcal{F}^{min}) = \mathcal{B}(\mathcal{F})\}. \tag{9}$$

Clearly we have $P(\mathcal{F}) \subseteq P^*(\mathcal{F})$.

A.2 Reducing to blocking families

Consider an arbitrary set family $\mathcal{F}$ over V. We may define its *upwards closure* by $\mathcal{F}^{\uparrow} = \{F' : F \subseteq F' \text{ for some } F \in \mathcal{F}\}$. In this section we argue that in order to solve a monotone optimization problem over sets in $\mathcal{F}$ it is often sufficient to work over its upwards-closure.

As already noted $\mathcal{B}(\mathcal{F}) = \mathcal{B}(\mathcal{F}^{\uparrow}) = \mathcal{B}(\mathcal{F}^{min})$ and hence one approach is via the blocking formulation $P^*(\mathcal{F}) = P^*(\mathcal{F}^{\uparrow})$. This requires two ingredients. First, we need a separation algorithm for the blocking relaxation, but indeed this is often available for many natural families such as spanning trees, perfect matchings, *st*-paths, and vertex covers. The second ingredient needed is the ability to turn an integral solution $\chi^{F'}$ from $P^*(\mathcal{F}^{\uparrow})$ or $P(\mathcal{F}^{\uparrow})$ into an integral solution $\chi^F \in P(\mathcal{F})$. We now argue that this is the case if a polytime separation algorithm is available for the blocking relaxation $P^*(\mathcal{F}^{\uparrow})$ or for the polytope $P(\mathcal{F})$.

For a polyhedron P, we denote its *dominant* by $P^{\uparrow} := \{z : z \geq x \text{ for some } x \in P\}$. The following observation is straightforward.

▶ **Claim 23.** *Let H be the set of vertices of the hypercube in $\mathbb{R}^V$. Then*

$$H \cap P(\mathcal{F}^{\uparrow}) = H \cap P(\mathcal{F})^{\uparrow} = H \cap P^*(\mathcal{F}^{\uparrow}).$$

In particular we have that $\chi^S \in P(\mathcal{F})^{\uparrow} \iff \chi^S \in P^*(\mathcal{F}^{\uparrow})$.

We use this observation to prove the following.

▶ **Lemma 24.** *Assume we have a separation algorithm for $P^*(\mathcal{F}^\uparrow)$. Then for any $\chi^S \in P^*(\mathcal{F}^\uparrow)$ we can find in polytime $\chi^M \in P(\mathcal{F})$ such that $\chi^M \leq \chi^S$.*

Proof. Let $S = \{1, 2, \ldots, k\}$. We run the following routine until no more elements can be removed:

For $i \in S$
  If $\chi^{S-i} \in P^*(\mathcal{F}^\uparrow)$ then $S = S - i$

Let χ^M be the output. We show that $\chi^M \in P(\mathcal{F})$. Since $\chi^M \in P^*(\mathcal{F}^\uparrow)$, by Claim 23 we know that $\chi^M \in P(\mathcal{F})^\uparrow$. Then by definition of dominant there exists $x \in P(\mathcal{F})$ such that $x \leq \chi^M \in P(\mathcal{F})^\uparrow$. It follows that the vector x can be written as $x = \sum_i \lambda_i \chi^{U_i}$ for some $U_i \in \mathcal{F}$ and $\lambda_i \in (0, 1]$ with $\sum_i \lambda_i = 1$. Clearly we must have that $U_i \subseteq M$ for all i, otherwise x would have a non-zero component outside M. In addition, if for some i we have $U_i \subsetneq M$, then there must exist some $j \in M$ such that $U_i \subseteq M - j \subsetneq M$. Hence $M - j \in \mathcal{F}^\uparrow$, and thus $\chi^{M-j} \in P(\mathcal{F})^\uparrow$ and $\chi^{M-j} \in P^*(\mathcal{F}^\uparrow)$. But then when component j was considered in the algorithm above, we would have had S such that $M \subseteq S$ and so $\chi^{S-j} \in P^*(\mathcal{F}^\uparrow)$ (that is $\chi^{S-j} \in P(\mathcal{F})^\uparrow$), and so j should have been removed from S, contradiction. ◀

We point out that for many natural set families $\mathcal{F}$ we can work with the relaxation $P^*(\mathcal{F}^\uparrow)$ assuming that it admits a separation algorithm. Then, if we have an algorithm which produces $\chi^{F'} \in P^*(\mathcal{F}^\uparrow)$ satisfying some approximation guarantee for a monotone problem, we can use Lemma 24 to construct in polytime $F \in \mathcal{F}$ which obeys the same guarantee.

Moreover, notice that for Lemma 24 to work we do not need an actual separation oracle for $P^*(\mathcal{F}^\uparrow)$, but rather all we need is to be able to separate over $0-1$ vectors only. Hence, since the polyhedra $P^*(\mathcal{F}^\uparrow)$, $P(\mathcal{F}^\uparrow)$ and $P(\mathcal{F})^\uparrow$ have the same $0-1$ vectors (see Claim 23), a separation oracle for either $P(\mathcal{F}^\uparrow)$ or $P(\mathcal{F})^\uparrow$ would be enough for the routine of Lemma 24 to work. We now show that this is the case if we have a polytime separation oracle for $P(\mathcal{F})$. The following result shows that if we can separate efficiently over $P(\mathcal{F})$ then we can also separate efficiently over the dominant $P(\mathcal{F})^\uparrow$.

▶ **Claim 25.** *If we can separate over a polyhedron P in polytime, then we can also separate over its dominant $P^\uparrow$ in polytime.*

Proof. Given a vector y, we can decide whether $y \in P^\uparrow$ by solving

$$\begin{aligned} x + s &= y \\ x &\in P \\ s &\geq 0. \end{aligned}$$

Since can we easily separate over the first and third constraints, and a separation oracle for P is given (i.e. we can also separate over the set of constraints imposed by the second line), it follows that we can separate over the above set of constraints in polytime. ◀

Now we can apply the same mechanism from Lemma 24 to turn feasible sets from $\mathcal{F}^\uparrow$ into feasible sets in $\mathcal{F}$.

▶ **Corollary 26.** *Assume we have a separation algorithm for $P(\mathcal{F})^\uparrow$. Then for any $\chi^S \in P(\mathcal{F})^\uparrow$ we can find in polytime $\chi^M \in P(\mathcal{F})$ such that $\chi^M \leq \chi^S$.*

We conclude this section by making the remark that if we have an algorithm which produces $\chi^{F'} \in P(\mathcal{F}^\uparrow)$ satisfying some approximation guarantee for a monotone problem, we can use Corollary 26 to construct $F \in \mathcal{F}$ which obeys the same guarantee.

B Convex relaxations for constrained submodular minimization

We will be working with upwards-closed set families $\mathcal{F}$, and their blocking relaxations $P^*(\mathcal{F})$. As we now work with arbitrary vectors $z \in [0,1]^n$, we must specify how our objective function $f(S)$ behaves on all points $z \in P^*(\mathcal{F})$. Formally, we call $g : [0,1]^V \to \mathbb{R}$ an *extension* of f if $g(\chi^S) = f(S)$ for each $S \subseteq V$. For a submodular objective function $f(S)$ there can be many extensions of f to $[0,1]^V$ (or to $\mathbb{R}^V$). The most popular one has been the so-called *Lovász Extension* (introduced in [30]) due to several of its desirable properties.

We present one of several equivalent definitions for the Lovász Extension. Let $0 < v_1 < v_2 < ... < v_m \leq 1$ be the distinct positive values taken in some vector $z \in [0,1]^V$. We also define $v_0 = 0$ and $v_{m+1} = 1$ (which may be equal to v_m). Define for each $i \in \{0, 1, ..., m\}$ the set $S_i = \{j : z_j > v_i\}$. In particular, S_0 is the support of z and $S_m = \emptyset$. One then defines

$$f^L(z) = \sum_{i=0}^{m} (v_{i+1} - v_i) f(S_i).$$

▶ **Lemma 27** (Lovász [30]). *The function f^L is convex if and only if f is submodular.*

One could now attack constrained submodular minimization by solving the problem

$$\text{(SA-LE)} \qquad \min f^L(z) : z \in P^*(\mathcal{F}), \tag{10}$$

and then seek rounding methods for the resulting solution. This is the approach used in [5, 20, 21]. We refer to the above as the *single-agent Lovász extension formulation*, abbreviated as (SA-LE).

B.1 Tractability of the single-agent formulation (SA-LE)

In this section we show that one may solve (SA-LE) approximately as long as a polytime separation algorithm for $P^*(\mathcal{F})$ is available. This is useful in several settings and in particular for our methods which rely on the multi-agent Lovász extension formulation (discussed in Section B.2).

Polytime Algorithms. One may apply the Ellipsoid Method to obtain a polytime algorithm which approximately minimizes a convex function over a polyhedron K as long as various technical conditions hold. For instance, one could require that there are two ellipsoids $E(a, A) \subseteq K \subseteq E(a, B)$ whose encoding descriptions are polynomially bounded in the input size for K. We should also have polytime (or oracle) access to the convex objective function defined over $\mathbf{R}^n$. In addition, one must be able to polytime solve the subgradient problem for f.[2] One may check that the subgradient problem is efficiently solvable for Lovász extensions of polynomially encodable submodular functions. We call f *polynomially encodable* if the values $f(S)$ have encoding size bounded by a polynomial in n (we always assume this for our functions). If these conditions hold, then methods from [15] imply that for any $\epsilon > 0$ we may find an approximately feasible solution for K which is approximately optimal. By approximate here we mean for instance that the objective value is within ϵ of the real optimum. This can be done in time polynomially bounded in n (size of input say) and $\log \frac{1}{\epsilon}$. Let us give a few details for our application.

[2] For a given y, find a subgradient of f at y.

Our convex problem's feasible space is $P^*(\mathcal{F})$ and it is easy to verify that our optimal solutions will lie in the $0-1$ hypercube H. So we may define the feasible space to be H and the objective function to be $g(z) = f^L(z)$ if $z \in H \cap P^*(\mathcal{F})$ and $= \infty$ otherwise. (Clearly g is convex in $\mathbf{R}^n$ since it is a pointwise maximum of two convex functions; alternatively, one may define the Lovász Extension on $\mathbf{R}^n$ which is also fine.) Note that g can be evaluated in polytime by the definition of f^L as long as f is polynomially encodable. We can now easily find an ellipsoid inside H and one containing H each of which has poly encoding size. We may thus solve the convex problem to within $\pm\epsilon$-optimality in time bounded by a polynomial in n and $\log \frac{1}{\epsilon}$.

▶ **Corollary 28.** *Consider a class of problems $\mathcal{F}, f$ for which f's are submodular and polynomially-encodable in $n = |V|$. If there is a polytime separation algorithm for the family of polyhedra $P^*(\mathcal{F})$, then the convex program (SA-LE) can be solved to accuracy of $\pm\epsilon$ in time bounded by a polynomial in n and $\log \frac{1}{\epsilon}$.*

B.2 The multi-agent formulation

The single-agent formulation (SA-LE) discussed above has a natural extension to the multi-agent setting. This was already introduced in [5] for the case $\mathcal{F} = \{V\}$.

$$\text{(MA-LE)} \qquad \min \sum_{i\in[k]} f_i^L(z_i) : z_1 + z_2 + \cdots + z_k \in P^*(\mathcal{F}). \tag{11}$$

We refer to the above as the *multi-agent Lovász extension formulation*, abbreviated as (MA-LE). We can solve (MA-LE) as long as we have polytime separation of $P^*(\mathcal{F})$. This follows the approach from the previous section (see Corollary 28) except our convex program now has k vectors of variables $z_1, z_2, \ldots, z_k$ (one for each agent) such that $z = \sum_i z_i$. This problem has the form $\{\min g(w) : w \in W \subseteq \mathbf{R}^{nk}\}$ where W is the full-dimensional convex body $\{w = (z_1, ..., z_k) : \sum_i z_i \in P^*(\mathcal{F})\}$ and $g(w) = g(z_1, z_2, \ldots, z_k) = \sum_{i\in[k]} f_i^L(z_i)$ is convex. Clearly we have a polytime separation routine for W, and hence we may apply Ellipsoid as in the single-agent case.

C Invariance under the lifting reduction

We prove some of the results from Table 1 in Section 3.1. For a subset of edges $S \subseteq E$ we define its *coverage* $cov(S)$ as the set of nodes $v \in V$ saturated by S. That is, $v \in cov(S)$ if there exists $i \in [k]$ such that $(i, v) \in S$. By definition of $\mathcal{F}'$ (see Section 3.1) it is straightforward that for each $S \subseteq E$ we have that

$$S \in \mathcal{F}' \iff cov(S) \in \mathcal{F} \text{ and } |S| = |cov(S)|. \tag{12}$$

For a set $S \subseteq E$, a set $B \subseteq S$ is called a *basis* of S if B is an inclusion-wise maximal independent subset of S. Our next result describes how bases and their cardinalities behave under the lifting reduction.

▶ **Lemma 29.** *Let S be an arbitrary subset of E. Then for any basis B (over $\mathcal{F}'$) of S there exists a basis B' (over $\mathcal{F}$) of $cov(S)$ such that $|B'| = |B|$. Moreover, for any basis B' of $cov(S)$ there exists a basis B of S such that $|B| = |B'|$.*

Proof. For the first part, let B be a basis of S and take $B' := cov(B)$. Since $B \in \mathcal{F}'$ we have by (12) that $B' \in \mathcal{F}$ and $|B'| = |B|$. Now, if B' is not a basis of $cov(S)$ then we can

find an element $v \in cov(S) - B'$ such that $B' + v \in \mathcal{F}$. Moreover, since $v \in cov(S)$ there exists $i \in [k]$ such that $(i, v) \in S$. But then we have that $B + (i, v) \subseteq S$ and $B + (i, v) \in \mathcal{F}'$, a contradiction with the fact that B was a basis of S.

For the second part, let B' be a basis of $cov(S)$. For each $v \in B'$ let i_v be such that $(i_v, v) \in S$, and take $B := \uplus_{v \in B'}(i_v, v)$. It is clear by definition of B that $cov(B) = B'$ and $|B| = |B'|$. Hence $B \in \mathcal{F}'$ by (12). If B is not a basis of S there exists an edge $(i, v) \in S - B$ such that $B + (i, v) \in \mathcal{F}'$. But then by (12) we have that $cov(B + (i, v)) \in \mathcal{F}$ and $B' \subsetneq cov(B + (i, v)) \subseteq cov(S)$, a contradiction since B' was a basis of $cov(S)$. ◀

We say that $(V, \mathcal{F})$ is a *p-system* if for each $U \subseteq V$, the cardinality of the largest basis of U is at most p times the cardinality of the smallest basis of U. The following result is a direct consequence of Lemma 29.

▶ **Proposition 30.** *If $(V, \mathcal{F})$ is a p-system, then $(E, \mathcal{F}')$ is a p-system.*

▶ **Corollary 31.** *If $\mathcal{F}$ corresponds to the set of bases of a p-system $(V, \mathcal{I})$, then $\mathcal{F}'$ also corresponds to the set of bases of some p-system $(E, \mathcal{I}')$.*

Proof. Consider $(E, \mathcal{I}')$ where $\mathcal{I}' := \{S \subseteq E : cov(S) \in \mathcal{I} \text{ and } |cov(S)| = |S|\}$. Then by Proposition 30 we have that $(E, \mathcal{I}')$ is a p-system. It is now straightforward to check that $\mathcal{F}'$ corresponds precisely to the set of bases of $(E, \mathcal{I}')$. ◀

The following two results follow from Proposition 30 and Corollary 31 and the fact that matroids are precisely the class of 1-systems.

▶ **Corollary 32.** *If $(V, \mathcal{F})$ is a matroid, then $(E, \mathcal{F}')$ is a matroid.*

▶ **Corollary 33.** *Assume $\mathcal{F}$ is the set of bases of some matroid $\mathcal{M} = (V, \mathcal{I})$, then $\mathcal{F}'$ is the set of bases of some matroid $\mathcal{M}' = (E, \mathcal{I}')$.*

Let the functions f_i and f be as described in the lifting reduction in Section 3.1. The following result establishes the relationship between $f^M(z_1, \ldots, z_k)$ and $\sum_{i \in [k]} f_i^M(z_i)$.

▶ **Lemma 34.** *Let the functions f_i and f be as described in the lifting reduction in Section 3.1. Then for any vector $\bar{z} = (z_1, z_2, \ldots, z_k) \in [0, 1]^E$, where $z_i \in [0, 1]^V$ is the vector associated with agent i, we have that $f^M(\bar{z}) = f^M(z_1, z_2, \ldots, z_k) = \sum_{i \in [k]} f_i^M(z_i)$.*

Proof. We use the definition of the multilinear extension in terms of expectations (see Section 3.2). Recall that for a vector $z \in [0, 1]^V$, R^z denotes a random set that contains element v_i independently with probability z_{v_i}. We use $\mathbb{P}_z(S)$ to denote $\mathbb{P}[R^z = S]$. We then have

$$\begin{aligned}
f^M(\bar{z}) &= \mathbb{E}[f(R^{\bar{z}})] = \sum_{S \subseteq E} f(S)\mathbb{P}_{\bar{z}}(S) \\
&= \sum_{S_1 \subseteq V} \sum_{S_2 \subseteq V} \cdots \sum_{S_k \subseteq V} [\sum_{i \in [k]} f_i(S_i)] \cdot \mathbb{P}_{(z_1, z_2, \ldots, z_k)}(S_1, S_2, \ldots, S_k) \\
&= \sum_{i \in [k]} \sum_{S_1 \subseteq V} \sum_{S_2 \subseteq V} \cdots \sum_{S_k \subseteq V} f_i(S_i) \cdot \mathbb{P}_{(z_1, z_2, \ldots, z_k)}(S_1, S_2, \ldots, S_k) \\
&= \sum_{i \in [k]} \sum_{S_i \subseteq V} f_i(S_i) \sum_{S_j \subseteq V, j \neq i} \mathbb{P}_{(z_1, z_2, \ldots, z_k)}(S_1, S_2, \ldots, S_k) \\
&= \sum_{i \in [k]} \sum_{S_i \subseteq V} f_i(S_i)\mathbb{P}_{z_i}(S_i) = \sum_{i \in [k]} \mathbb{E}[f_i(S_i^{z_i})] = \sum_{i \in [k]} f_i^M(z_i).
\end{aligned}$$

◀

Generalized Assignment of Time-Sensitive Item Groups

Kanthi Sarpatwar
IBM Research, Yorktown Heights, NY, USA
sarpatwa@us.ibm.com

Baruch Schieber
IBM Research, Yorktown Heights, NY, USA
sbar@us.ibm.com

Hadas Shachnai
Computer Science Department, Technion, Haifa, Israel
hadas@cs.technion.ac.il

Abstract

We study the generalized assignment problem with time-sensitive item groups (χ-AGAP). It has central applications in advertisement placement on the Internet, and in virtual network embedding in Cloud data centers. We are given a set of items, partitioned into n groups, and a set of T identical bins (or, time-slots). Each group $1 \leq j \leq n$ has a time-window $\chi_j = [r_j, d_j] \subseteq [T]$ in which it can be packed. Each item i in group j has a size $s_i > 0$ and a non-negative utility u_{it} when packed into bin $t \in \chi_j$. A bin can accommodate at most one item from each group and the total size of the items in a bin cannot exceed its capacity. The goal is to find a feasible packing of a subset of the items in the bins such that the total utility from groups that are *completely* packed is maximized. Our main result is an $\Omega(1)$-approximation algorithm for χ-AGAP. Our approximation technique relies on a non-trivial rounding of a configuration LP, which can be adapted to other common scenarios of resource allocation in Cloud data centers.

2012 ACM Subject Classification Theory of computation → Packing and covering problems

Keywords and phrases Approximation Algorithms, Packing and Covering problems, Generalized Assignment problem

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.24

Acknowledgements H. Shachnai's work was conducted during a visit to DIMACS partially supported by the National Science Foundation under grant number CCF-1445755.

1 Introduction

In the classic *generalized assignment problem* (GAP), we are given a set of N items and T bins, $[T] = \{1, 2, \ldots, T\}$. Each item $i \in [N]$ has a size $s_i > 0$ and a non-negative utility u_{it} when packed into a bin $t \in [T]$.[1] The goal is to feasibly pack in the bins a subset of the items of maximum total utility. GAP has been widely studied, with applications ranging from manufacturing systems to regional planning (see, e.g., [3, 10]). In discrete optimization, GAP is well-known as a special case of the *separable assignment* problem and, more generally, of *submodular maximization* (see, e.g., [27, 15, 5, 6]). We consider an *all-or-nothing* variant of

[1] While item sizes may also depend on the bins, we focus here on the uniform case arising in our applications.

GAP, whose central applications include targeted advertising and virtual network embedding in Cloud data centers.

In *all-or-nothing* generalized assignment with time windows (χ-AGAP) we are given a set of T identical bins (or time-slots) and a set I of N items partitioned into n disjoint groups J. Each group $j \in J$ consists of a subset of items $I_j \subseteq I$ and has a time-window $\chi_j = [r_j, d_j] \subseteq [T]$ in which it can be packed. An item $i \in [N]$ which belongs to group j has a size $s_i > 0$ and a non-negative utility u_{it} when packed into bin $t \in \chi_j$. Each bin can accommodate at most one item from each group and the total size of the items in a bin cannot exceed its capacity. The goal is to find a feasible packing of a subset of the items in the bins such that the total utility from groups that are *completely packed* is maximized.

As a primary motivation for our study, consider the management of an advertising campaign (i.e., a series of advertisements sharing a single theme) targeted at specific audience. Given such a large set of campaigns that can potentially be delivered to the media audience, a service provider attempts to fully deliver a subset of campaigns that maximizes the total revenue [26, 17, 21]. Each campaign is associated with a time-window (certain days/weeks, or hours in a given day) during which all of its ads need to be delivered. To increase the number of viewers exposed to an ad campaign, any continuous posting of ads (e.g., commercial break on TV) may contain a *single* ad from this campaign. Each ad has a given length (= size), which remains the same, regardless of the time slot in which it is posted, and possibly different revenues when placed in different breaks. The revenue from an ad campaign is the sum of the individual revenues of its ads, that is gained only if all of the campaign ads were assigned to breaks. Hence, a campaign scheduling instance can be cast as an instance of χ-AGAP.

Our problem has a central application also in *virtual network embedding (VNE)* in Cloud data centers, where a collection of virtual networks is mapped to a substrate *software defined network (SDN)* [28, 23, 20]. In the initial step of the VNE process, the nodes of each virtual network are mapped to nodes in the physical network. Each virtual network can be viewed as a *request* that is fulfilled only if all of its demands can be satisfied; else, the request is dropped. A common approach is to assign the nodes belonging to a single request to different substrate nodes and interconnect them by physical paths in the substrate network corresponding to the virtual links [9, 24]. Each node of a virtual network is associated with a resource demand (= size) and a revenue from assigning the node to a given physical node (= bin). The goal is to maximize the revenue gained from the VNE process. This yields an instance of χ-AGAP, where all groups (= virtual networks) share the same time window.

1.1 Related Work

Given an algorithm $\mathcal{A}$, let $\mathcal{A}(I), OPT(I)$ denote the utility of the solution output by $\mathcal{A}$ and by an optimal solution for a problem instance I, respectively. For $\rho \in (0, 1]$, we say that $\mathcal{A}$ is a ρ-approximation algorithm if, for any instance I, $\frac{\mathcal{A}(I)}{OPT(I)} \geq \rho$.

Recall that the special case of χ-AGAP where all groups consist of a *single* item and have the same time window $[T]$ yields an instance of GAP, where each item takes a single size over all bins. GAP is known to be APX-hard already in this case, even if there are only two possible item sizes, each item can take two possible profits, and all bin capacities are identical [7]. Fleischer et al. [15] obtained a $(1 - e^{-1})$-approximation for GAP, as a special case of the *separable assignment problem*. The best known ratio is $1 - e^{-1} + \varepsilon$ for some absolute constant $\varepsilon > 0$ [14].

Adany *et al.* [1] considered *all-or-nothing* GAP (AGAP), the special case of χ-AGAP where $\chi_j = [T]$ for all $j \in J$. They presented an $\Omega(1)$-approximation algorithm for general

instances, and a $(1/3-\varepsilon)$-approximation for the special case where the utility of an item is identical across the bins, called the *group packing* (GP) problem. They also showed that unless $NP = ZPP$, GP cannot be approximated within a polynomial in the minimum size of a group, if we allow forbidden bins for the items. This implies that a polynomial time constant approximation for χ-AGAP with arbitrary forbidden bins is unlikely to exist. Indeed, our results rely strongly on the structural properties of the interval graph corresponding to the time windows which define the set of allowed bins for each group.

The Ad placement problem, introduced by Adler et al. [2], is the special case of χ-AGAP where all items in group $j \in J$ are identical, and each item has the same utility across the bins; also, all groups j share the same time window $\chi_j = [T]$. Freund and Naor [16] presented a $(1/3-\varepsilon)$-approximation for the problem. Ad placement has also been widely studied under different objectives (see, e.g., [12, 16, 13, 19, 18] and the comprehensive survey in [22]). In [25], we present approximation algorithms for more general scenarios of Ad placement, where groups may have different time windows, or ads are associated with multiple resource demands.

A related problem is *group packing of items into multiple knapsacks* (GMKP), introduced in [8]. In this generalization of *multiple knapsack* groups of items need to be placed in a set of bins. As in χ-AGAP, a revenue is gained for a group only if all of its items are packed. However, GMKP differs from χ-AGAP in several ways: (i) two items from the same group can be placed in the same bin. (ii) the utility of each item is the same across the bins, and (iii) all groups share the same time window, $[T]$. The paper [8] presents approximation algorithms for GMKP, assuming that the total size of the items in group j satisfies $\sum_{i\in I_j} s_i \leq \delta T$, for varying values of $\delta \in (0,1)$.

Our problem relates also to the *hypermatching assignment problem (HAP)* introduced in [11]. The input is a set of clients and a set of bundles of items. Each client has a budget; each bundle has a price and a profit depending on the client to which the bundle is assigned. Clients wish to buy one or more disjoint bundles of items . The goal is to maximize the total profit, while respecting client budgets. We can view the bundles as *groups*, and the clients as *time slots*, but then the remaining constraints are fairly different.

1.2 Contributions and Techniques

Our main results are constant factor approximation algorithms for χ-AGAP. Let $\chi_j = [r_j, d_j]$ denote the time window for processing group $j \in J$, and let $|\chi_j| = d_j - r_j + 1$ be the length of this interval. Throughout the paper, we assume that the time windows are large enough, namely, there is a constant $\lambda \in (0,1)$, such that $|I_j| \leq \lambda|\chi_j|$ for any group of items j. Such an assumption is quite reasonable in scenarios arising in our applications. We call λ the *slackness* parameter of the instance.

In Section 3, we present an $\Omega(1)$-approximation algorithm for *laminar* instances of χ-AGAP,[2] assuming that $\lambda < \frac{1}{5}$. We then extend the algorithm (in Section 4) to obtain an $\Omega(1)$-approximation for general χ-AGAP instances, assuming that $\lambda < \frac{1}{20}$. Thus, a main contribution of this paper is a general framework for obtaining a constant factor approximation based on the slackness of a given instance.

Techniques. Adany *et al.* [1] obtained a constant factor approximation algorithm for the special case where $\chi_j = [T] = \{1, 2, \ldots, T\}$, for all $j \in J$, assuming $\lambda < \frac{1}{2}$. We note that

[2] See the formal definition in Section 2.

even this special case is non-trivial and was solved via a novel reduction to maximizing a submodular function subject to a matroid constraint (over an exponential size universe of elements). An attempt to apply the same technique to the more general problem fails, due to the polynomially many knapsack constraints (representing the time windows for the n groups).

Thus, we apply a different approximation technique, which relies on a non-trivial rounding of the (fractional) solution for a configuration LP that guarantees a partial packing of a subset of groups.[3] This packing satisfies certain conditions, while still having a total utility that is within a constant factor from the optimal. Subsequently, using an elaborate evicting and repacking phase that exploits the structural properties of the instance, we obtain a feasible packing of *all* items in these groups.

We use our algorithm for laminar instances as a building block for solving general instances, via a transformation of a general instance to laminar. While the transformation is simple, applying the algorithm to the resulting laminar instance requires new ideas, as we must take into consideration the potential utility of each item when packed in its *original* time window (see Section 4). For the analysis we prove a general Repacking Theorem (see Section 3) stating the conditions required for obtaining a feasible solution from a *partial* packing of a given set of groups. This general theorem may be of independent interest in solving *all-or-nothing* variants of packing problems on interval graphs.

2 Preliminaries

We start with some definitions and notation. An instance of χ-AGAP consists of a set of identical (unit-sized) bins and a set of items I; each item $i \in I$ has a size $s_i > 0$ and utility u_{it} when packed into bin $t \in [T]$. The items are partitioned into n disjoint groups J. Each group $j \in J$ contains a subset of items $I_j \subseteq I$ which can be packed in the time window $\chi_j = [r_j, d_j] \subseteq T$.

A *feasible* packing of the subset $J' \subseteq J$ is an assignment of $|I_j|$ bins to every group $j \in J'$ such that the total size of items packed in any bin $t \in [T]$ is at most 1. Formally, $p : I' \to [T]$ is a feasible packing of the items in $I' = \bigcup_{j \in J'} I_j$ into bins if

1. For any $I_j \subseteq I'$, $|\bigcup_{i \in I_j} p(i)| = |I_j|$, i.e., the items in I_j are assigned to distinct bins.
2. For all $i \in I_j \subseteq I'$, $p(i) \in \chi_j$,
3. For all $t \in [T]$, $\sum_{\{i \in I' | p(i)=t\}} s_i \leq 1$.

The total utility of the packing p is given by $U_p = \sum_{j \in J'} \sum_{i \in I_j} u_{ip(i)}$. We say that $\lambda \in (0, 1)$ is the *slackness* parameter of a given instance if $|I_j| \leq \lambda |\chi_j|$ for all $j \in J$. Also, let $a_j = \sum_{i \in I_j} s_i$ denote the total size of items in group $j \in J$.

Finally, a set of intervals is *laminar* if for any two intervals χ_1 and χ_2, exactly one of the following holds: $\chi_1 \subseteq \chi_2$, $\chi_2 \subset \chi_1$ or $\chi_1 \cap \chi_2 = \emptyset$.

3 Approximation Algorithm for Laminar Instances

We first consider the case where $\mathcal{L} = \{\chi_j : j \in J\}$ forms a laminar family of intervals.

[3] Our rounding technique bears some similarities to the randomized rounding with alterations approach of [4].

Overview. Our algorithm proceeds in two phases. In the first phase, we find a subset of groups J' and items $I' \subseteq \bigcup_{j \in J'} I_j$ from these groups, along with a packing $p : I' \to [T]$ satisfying the following properties:

(1) For some constant $\alpha \in (0,1)$, for all $\chi \in \mathcal{L}$: $\sum_{j \in J' : \chi_j \subseteq \chi} a_j \leq \alpha|\chi|$. That is, the total size of *chosen* groups with time windows contained in χ is at most $\alpha|\chi|$.

(2) For any bin $t \in [T]$, the total size of items packed into t is at most 1, i.e., $\sum_{i \in I' : p(i)=t} s_i \leq 1$.

(3) Let OPT be the total profit of an optimal solution. Then, for some constant $\beta \in (0,1)$, $\sum_{i \in I'} u_{ip(i)} \geq \beta OPT$.

We note that, in this phase, some of the items in J' may not be packed into bins. The algorithm then moves to an *evict and repack* phase. Starting with a partial solution obtained from the first phase, we empty some of the bins. The eviction is performed in such a way that, for any interval $\chi \in \mathcal{L}$, at least a constant fraction of bins in χ are empty. This eviction step, which results in a constant factor loss in the approximation ratio, will facilitate the repacking of *all* items in the groups J'.

Phase 1. We start by solving a *linear programming (LP)* relaxation of our problem.

Configuration LP. For any group $j \in J$, a valid configuration M is an assignment of the items in group j into $|I_j|$ bins in χ_j. We can view a configuration for group j as a mapping $M : I_j \to \chi_j$, where $M(i)$ indicates the bin to which $i \in I_j$ is assigned. For a given group $j \in J$, let $\mathcal{C}_j$ be the set of all such valid configurations containing the items of group j. For a given bin $t \in [T]$, denote by $\mathcal{C}_t$ the set of valid configurations M in which there exists $i \in I$ such that $M(i) = t$. Define $\mathcal{C} = \bigcup_{j \in J} \mathcal{C}_j = \bigcup_{t \in [T]} \mathcal{C}_t$. Let x_M denote the indicator variable for choosing a configuration M. Throughout the discussion, each configuration M is (implicitly) associated with the assignment of the items of specific group $1 \leq j \leq n$. The utility of a configuration $M \in \mathcal{C}_j$ is given by $u_M = \sum_{i \in I_j} u_{iM(i)}$. Finally, for a given configuration, M, $i = M^{-1}(t)$ if i is assigned in M to bin t. The configuration LP can be stated as follows.

$$\text{Maximize} \left\{ \sum_{M \in \mathcal{C}} u_M x_M : \sum_{M \in \mathcal{C}_t} x_M s_{M^{-1}(t)} \leq 1 \, (\forall t \in [T]), \ \sum_{M \in \mathcal{C}_j} x_M \leq 1 \, (\forall j \in J), \ x_M \geq 0 \right\}$$

The first constraint ensures that the total size of items packed into a bin does not violate its capacity, and the second constraint implies that we choose at most one configuration for each group j. Note that the LP has exponentially many variables, and therefore we solve it using a dual separation oracle. The dual of the configuration LP is:

$$\text{Minimize} \left\{ \sum_{t \in [T]} y_t + \sum_{j \in J} z_j : z_j + \sum_{t \in \chi_j : M \in \mathcal{C}_t} y_t s_{M^{-1}(t)} \geq u_M \, (\forall j \in J, \ M \in \mathcal{C}_j), \ y_t, \, z_j \geq 0 \right\}$$

The dual program has exponentially many constraints. However, as shown below, it admits a separation oracle; thus, using the ellipsoid algorithm, it can be solved in polynomial time. Consider a dual solution $(y_t : t \in [T], \, z_j : j \in J)$. For each group $j \in J$, we construct a complete bipartite graph $H_j = (I_j, \chi_j, E_j)$, where $E_j = \{(i,t) : i \in I_j, t \in \chi_j\}$. For each item $i \in I_j$ and bin $t \in \chi_j$ we set $w(i,t) = u_{it} - y_t s_i$. Now, to test if there is a violating constraint corresponding to a configuration M for group j, we compute a maximum weight matching in H_j, denoted by M_j^*, in which every item $i \in I_j$ is matched.[4] Clearly, if there

[4] Since the edge weights in H_j may be negative, this can be done by a standard shift of edge weights, namely, setting $w'(i,j) = w(i,j) + |I_j|W$, where $W = \max_{(i,j) \in E_j} w(i,j)$, and solving maximum weight matching w.r.t. w'.

is a violating constraint corresponding to j, then the total weight for M_j^* has to satisfy $\sum_{(i,t)\in M_j^*} w(i,j) > z_j$.

Rounding Algorithm. Let $\pi_1, \pi_2 \in (0,1)$ be some constants (to be determined). We first solve the configuration LP to obtain an optimal fractional solution $(x^*_M : M \in \mathcal{C})$. Let $X_j = \sum_{M\in\mathcal{C}_j} x^*_M$. For each interval $\chi \in \mathcal{L}$, we associate a knapsack capacity of $\pi_2|\chi|$. We construct an ordering of groups satisfying the following property. If j and ℓ are two groups such that $\chi_j \subseteq \chi_\ell$, then j appears before ℓ in $\mathcal{O}$. In other words, we process groups in a *bottom up fashion* with respect to $\mathcal{L}$.

While considering a group j, we check if the knapsack capacity corresponding to χ_j has been violated. Formally, let J' denote the set of groups chosen before j; then, if $\sum_{j'\in J':\chi_{j'}\subseteq\chi_j} a_{j'} \le \pi_2|\chi_j|$, we add j to J' with probability $\pi_1 \sum_{M\in\mathcal{C}_j} x^*_M = \pi_1 X_j$; otherwise we discard j. Note that, to simplify the performance analysis, the knapsack constraint is checked *before* j is added, therefore the knapsack capacity may be slightly violated.

Let $I' \subseteq \cup_{j\in J'} I_j$ be the set of items in the groups of J' that have been packed so far, and suppose that j has been chosen, i.e., $j \in J'$. Now, we choose exactly one configuration for j with probability $\frac{x^*_M}{X_j}$. Denote the chosen configuration by M_j. For each $(i,t) \in M_j$, we now examine the total size of items packed in bin t. If $\sum_{i'\in I':p(i')=t} s_{i'} \le 1$ then we pack item i into bin t, i.e., $p(i) = t$. Note that this may result in violation of the capacity constraints in some bins. Thus, after processing all groups, we examine bins with violated capacities. Denote by S_t the set of currently packed items in such a bin t, and let i_t denote the last item packed into t. Before adding i_t, the capacity constraint was not violated; thus, $S_t \setminus \{i_t\}$ can be feasibly packed into t. We evict i_t from t if the combined profit of all the items in $S_t \setminus \{i_t\}$ is higher than that of i_t; otherwise, we evict all items except i_t from bin t. The pseudocode for the above rounding scheme is given in Algorithm 1.

Denote the *mean* (or *expected value*) of a random variable X by $\mathbb{E}[X]$, and the probability that an event E occurs by $\mathbb{P}(E)$. Note that Algorithm 1 is randomized; therefore, J', I' and p (and any function involving them) are random variables. The next lemma states some properties of the packing obtained in Phase 1.

▶ **Lemma 1.** *For suitable constants $0 < \pi_1 < \pi_2 < 1$ and $\lambda \in (0,1)$, assuming that for any group $j \in J$, $|I_j| \le \lambda|\chi_j|$, the following hold for Algorithm 1:*

(i) *The expected utility of the items in I' is at least a constant fraction of the optimal LP value, i.e.,*

$$\mathbb{E}\left[\sum_{i\in I'} u_{ip(i)}\right] \ge \frac{\pi_1}{2}\left(1 - \left(\frac{\pi_1}{\pi_2} + \pi_1\right)\right)\left[\sum_{M\in\mathcal{C}} u_M x^*_M\right]. \tag{1}$$

(ii) *For any interval $\chi \in \mathcal{L}$, the total size of groups chosen in χ is at most a constant fraction of $|\chi|$ i.e.,*

$$\sum_{j\in J':\chi_j\subseteq\chi} a_j \le (\pi_2 + \lambda)|\chi|. \tag{2}$$

(iii) *For any bin $t \in T$, the total size of items packed into t is within its capacity, i.e.,*

$$\sum_{i\in I':p(i)=t} s_i \le 1. \tag{3}$$

Note that properties (ii) and (iii) hold unconditionally (i.e., with probability 1).

Algorithm 1 Rounding the configuration LP solution.

Input: Laminar χ-AGAP instance: $(I, J, T, \mathcal{L})$
Output: $J' \subseteq J$, $I' \subseteq \bigcup_{j \in J'} I_j$ and $p : I' \to [T]$

1: Initialize $J' \leftarrow \emptyset$ and $I' \leftarrow \emptyset$. Solve the configuration LP and obtain an optimal fractional solution $(x^*_M : M \in \mathcal{C})$
2: Let $\mathcal{O}$ be an ordering of groups J as follows: for any $j, \ell \in J$, if $\chi_j \subseteq \chi_\ell$ then j appears before ℓ in $\mathcal{O}$.
3: **for all** j in the ordered list $\mathcal{O}$ **do**
4: Let $X_j = \sum_{M \in \mathcal{C}_j} x^*_M$.
5: If $\sum_{j' \in J' : \chi_{j'} \subseteq \chi_j} a_{j'} \leq \pi_2 |\chi_j|$ set $J' \leftarrow J' \cup \{j\}$ with probability $\pi_1 X_j$.
6: **if** $j \in J'$ **then**
7: Let $\mathcal{C}^+_j$ denote the configurations $M \in \mathcal{C}_j$ such that $x^*_M > 0$.
8: Choose exactly one configuration, $M_j \in \mathcal{C}^+_j$ with probability $\frac{x^*_M}{X_j}$ for configuration M.
9: For all $(i, t) \in M_j$, if $\sum_{i' \in I' : p(i') = t} s_{i'} \leq 1$, set $I' \leftarrow I' \cup \{i\}$ and $p(i) = t$.
10: **end if**
11: **end for**
12: **for all** $t \in [T]$ **do**
13: **if** $\sum_{i' \in I' : p(i') = t} s_{i'} > 1$ **then** ▷ *handle violated bins*
14: Let i be the last item with $p(i) = t$. If $u_{it} \geq \sum_{i' \neq i : p(i') = t} u_{i't}$: set $p(i') = \emptyset$ and $I' \leftarrow I' \setminus \{i'\}$ $\forall i' \neq i$; otherwise, set $p(i) = \emptyset$ and $I' \leftarrow I' \setminus \{i\}$.
15: **end if**
16: **end for**

Proof. We start by lower bounding the expected utility obtained by using Algorithm 1 before eviction in Step 12.
(i) For any $t \in [T]$, let $U_t = \sum_{M \in \mathcal{C}} x^*_M \sum_{i \in I : (i,t) \in M} u_{it}$ denote the contribution of bin t to the optimal fractional solution value. It follows that $\sum_{t \in [T]} U_t = \sum_{M \in \mathcal{C}} x^*_M u_M$. Fix $t \in [T]$ and $i \in I$, and let j be the group containing item i. Let $\mathcal{C}_{i \to t}$ denote the set of all configurations M with $M(i) = t$. Let $\mathsf{iter}(j)$ be the iteration at which j is processed in Algorithm 1 (Step 3). We denote by $\mathbf{E}_{i \to t}$ the event "item i is assigned to bin t", i.e., $p(i) = t$. To show (1), it suffices to lower bound

$$\mathbb{E}\left[\sum_{i \in I' : p(i) = t} u_{it}\right] = \sum_{i \in I'} u_{it} \mathbb{P}(\mathbf{E}_{i \to t}),$$

for each $t \in [T]$. To this end, we define the following events:
($\mathbf{E}_1$) At the beginning of $\mathsf{iter}(j)$, $\sum_{j' \in J' : \chi_{j'} \subseteq \chi_j} a_{j'} \leq \pi_2 |\chi_j|$.
($\mathbf{E}_2$) Group j is added to J'.
($\mathbf{E}_3$) At the beginning of $\mathsf{iter}(j)$, $\sum_{i' \in I' : p(i') = t} s_{i'} \leq 1$.
We start by upper bounding the values $\mathbb{P}(\overline{\mathbf{E}_1})$ and $\mathbb{P}(\overline{\mathbf{E}_3})$. Since the probability of choosing a group j' is at most $\pi_1 X_{j'}$, using linearity of expectation, we have

$$\mathbb{E}\left[\sum_{j' \in J' : \chi_{j'} \subseteq \chi_j} a_{j'}\right] \leq \sum_{j' \in J : \chi_{j'} \subseteq \chi_j} \mathbb{P}(j' \in J') a_{j'} \leq \sum_{j' \in J : \chi_{j'} \subseteq \chi_j} \pi_1 X_{j'} a_{j'} \leq \pi_1 |\chi_j|. \tag{4}$$

The last inequality follows from the constraints of the linear program, i.e., the total fractional size of groups packed into interval χ_j is at most $|\chi_j|$. Combining (4) with Markov's inequality,

it follows that

$$\mathbb{P}(\overline{\mathbf{E}_1}) = \mathbb{P}\left(\sum_{j' \in J' : \chi_{j'} \subseteq \chi_j} a_{j'} > \pi_2 |\chi_j|\right) \leq \frac{\pi_1}{\pi_2}. \tag{5}$$

Similarly, we have $\mathbb{E}\left[\sum_{i' \in I' : p(i') = t} s_{i'}\right] \leq \sum_{M:(i',t) \in M} \pi_1 x^*_M \leq \pi_1$; thus, by Markov's inequality,

$$\mathbb{P}(\overline{\mathbf{E}_3}) = \mathbb{P}\left(\sum_{i' \in I' : p(i') = t} s_{i'} > 1\right) \leq \pi_1. \tag{6}$$

We have the following conditional probability inequalities:

$$\mathbb{P}(\mathbf{E}_2 | \mathbf{E}_1) = \pi_1 X_j \tag{7}$$

$$\mathbb{P}(\mathbf{E}_{i \to t} | \mathbf{E}_1 \cap \mathbf{E}_2 \cap \mathbf{E}_3) = \frac{\sum_{M \in \mathcal{C}_{i \to t}} x^*_M}{X_j} \tag{8}$$

Using (5) and (6),

$$\mathbb{P}(\mathbf{E}_1 \cap \mathbf{E}_3) = 1 - \mathbb{P}(\overline{\mathbf{E}_1} \cup \overline{\mathbf{E}_3}) \geq 1 - (\mathbb{P}(\overline{\mathbf{E}_1}) + \mathbb{P}(\overline{\mathbf{E}_3})) \geq 1 - (\frac{\pi_1}{\pi_2} + \pi_1) \tag{9}$$

Now, we compute the probability of $\mathbf{E}_{i \to t}$ as follows:

$$\begin{aligned}
\mathbb{P}(\mathbf{E}_{i \to t}) &= \mathbb{P}(\mathbf{E}_{i \to t} | \mathbf{E}_1 \cap \mathbf{E}_2 \cap \mathbf{E}_3) \mathbb{P}(\mathbf{E}_1 \cap \mathbf{E}_2 \cap \mathbf{E}_3) \\
&= \mathbb{P}(\mathbf{E}_{i \to t} | \mathbf{E}_1 \cap \mathbf{E}_2 \cap \mathbf{E}_3) \mathbb{P}(\mathbf{E}_2 | \mathbf{E}_1 \cap \mathbf{E}_3) \mathbb{P}(\mathbf{E}_1 \cap \mathbf{E}_3) \\
&= \mathbb{P}(\mathbf{E}_{i \to t} | \mathbf{E}_1 \cap \mathbf{E}_2 \cap \mathbf{E}_3) \mathbb{P}(\mathbf{E}_2 | \mathbf{E}_1) \mathbb{P}(\mathbf{E}_1 \cap \mathbf{E}_3)
\end{aligned}$$

The last equality follows from the fact that $\mathbb{P}(\mathbf{E}_2 | \mathbf{E}_1 \cap \mathbf{E}_3) = \mathbb{P}(\mathbf{E}_2 | \mathbf{E}_1)$. To see this, we note that by virtue of our algorithm $\mathbf{E}_2$ and $\mathbf{E}_3$ are conditionally independent events, given $\mathbf{E}_1$. Thus $\mathbb{P}(\mathbf{E}_2 | \mathbf{E}_1 \cap \mathbf{E}_3) = \mathbb{P}(\mathbf{E}_2 \cap \mathbf{E}_3 | \mathbf{E}_1) / \mathbb{P}(\mathbf{E}_3 | \mathbf{E}_1) = \mathbb{P}(\mathbf{E}_2 | \mathbf{E}_1)$. By the algorithm, $\mathbb{P}(\mathbf{E}_2 | \mathbf{E}_1) = \pi_1 X_j$. Thus, using (8) and (9), we have

$$\begin{aligned}
\mathbb{P}(\mathbf{E}_{i \to t}) &\geq \pi_1 X_j \left(\frac{\sum_{M \in \mathcal{C}_{i \to t}} x^*_M}{X_j}\right) \left(1 - \left(\frac{\pi_1}{\pi_2} + \pi_1\right)\right) \\
&= \pi_1 \left(\sum_{M \in \mathcal{C}_{i \to t}} x^*_M\right) \left(1 - \left(\frac{\pi_1}{\pi_2} + \pi_1\right)\right)
\end{aligned}$$

Thus, the expected utility for bin t is given by

$$\begin{aligned}
\mathbb{E}\left[\sum_{i \in I' : p(i) = t} u_{it}\right] &\geq \pi_1 \sum_{i \in I' : p(i) = t} u_{it} \left(\sum_{M \in \mathcal{C}_{i \to t}} x^*_M\right) \left(1 - \left(\frac{\pi_1}{\pi_2} + \pi_1\right)\right) \\
&= \pi_1 \left(1 - \left(\frac{\pi_1}{\pi_2} + \pi_1\right)\right) \left[\sum_{M \in \mathcal{C}} \sum_{i \in I : M(i) = t} u_{it} x^*_M\right]
\end{aligned}$$

Algorithm 2 Eviction Procedure.

Input: Packing $p : I' \to [T]$, a parameter $\pi_3 \in (0, 1)$
Output: Set of evicted bins E and a packing $p' : I'' \to [T] \setminus E$

Mark each interval $\chi \in \mathcal{L}$ as *unprocessed*. Initialize the set of evicted bins: $E \leftarrow \emptyset$, and let $I'' = I'$.
while $\exists$ a minimal unprocessed interval χ **do**
  if $|E \cap \chi| < \lceil (1 - \pi_3)|\chi| \rceil$ **then**
    Sort the bins $\chi \setminus E$ in ascending order of the total profit of items packed in them.
    Let E_χ denote the first $\tau = \lceil (1 - \pi_3)|\chi| \rceil - |E \cap \chi|$ bins in the above order.
    for all bins $t \in E_\chi$ **do**
      Evict the items in bin t, i.e., delete them from I''. Set $E \leftarrow E \cup \{t\}$.
    end for
    Mark χ as *processed*.
  end if
end while
Set $p'(i) = p(i)$, for all $i \in I''$
return E and p'

Finally, by the linearity of expectation,

$$\begin{aligned}
\mathbb{E}\left[\sum_{i \in I'} u_{ip(i)}\right] &= \sum_{t \in [T]} \mathbb{E}\left[\sum_{i \in I' : p(i) = t} u_{it}\right] \\
&\geq \sum_{t \in [T]} \pi_1 \left(1 - \left(\frac{\pi_1}{\pi_2} + \pi_1\right)\right) \left[\sum_{M \in \mathcal{C}} \sum_{i \in I : M(i) = t} u_{it} x^*_M\right] \\
&= \pi_1 \left(1 - \left(\frac{\pi_1}{\pi_2} + \pi_1\right)\right) \left[\sum_{M \in \mathcal{C}} u_M x^*_M\right]
\end{aligned}$$

Now, for each bin t, let i_t be the last item, and S_t the set of all items packed into t. If the capacity of t has been violated, we note that each of the sets $S_t \setminus i_t$ and $\{i_t\}$ can be feasibly packed into t. We evict either i_t or $S_t \setminus i_t$, whichever is less profitable and thereby lose an additional factor of $\frac{1}{2}$ in the worst case.
The proofs of parts (ii) and (iii) are deferred to the Appendix. ◀

Phase 2. We now show that for suitable values of the parameters $0 < \pi_1 < \pi_2 < 1$, we can obtain a packing of *all* the items in J', such that the total profit is an $\Omega(1)$ fraction of the optimum. This yields a constant approximation for laminar χ-AGAP. Let $\pi_3 \in [2\lambda, 1)$ (to be determined).
We first *evict* the items from some "low" profit bins, such that for any interval $\chi \in \mathcal{L}$, there are at least $(1 - \pi_3)|\chi|$ empty bins. (Recall that each interval χ can be viewed as a set of $|\chi|$ unit-sized bins.) Furthermore, the profit of items packed into bins that are not evicted is at least an $\Omega(1)$ fraction of the optimal solution value. Subsequently, we use the empty bins to repack all the unpacked items in the groups J'.
Eviction. Our eviction scheme is formally described in Algorithm 2. Bins are emptied while processing intervals in a *bottom-up fashion*, i.e., before an interval $\chi \in \mathcal{L}$ is processed, every sub-interval $\chi' \in \mathcal{L}$ where $\chi' \subset \chi$ is processed. Thus, in the main loop of the algorithm, we consider an unprocessed interval χ of minimum length in any given iteration (Step 2

Algorithm 3 Packing an item $i \in \hat{I}_j$.

1: If there exists a gray bin in *avail*(j) **goto** Step 2; otherwise, pick any white bin t in *avail*(j) and pack i into t. Color t gray and **exit**. If no such white bin exists, report **fail**.
2: Let t be a gray bin in *avail*(j), and S_t the set of items packed into t. If $\sum_{i' \in S_t} s_{i'} + s_i \leq 1$, then pack i into t and **exit**; otherwise, **goto** Step 3.
3: Pick a white bin t' in *avail*(j) and pack i into t'. Color t and t' as black and pair up $t \leftrightarrow t'$. If no such white bin exists, report **fail**.

of Algorithm 2). Denote the current set of all evicted bins by E. We check if the total number of evicted bins contained in χ satisfies $|\chi \cap E| \geq \lceil (1-\pi_3)|\chi| \rceil$. If not, we sort the bins $t \in \chi \setminus E$ in non-decreasing order of the total utility of items packed into them, given by $U_t = \sum_{\{i \in I' : p(i) = t\}} u_{it}$. We evict the first $\tau = \lceil (1-\pi_3)|\chi| \rceil - |\chi \cap E|$ bins from $\chi \setminus E$.

The following lemma shows that at the end of this procedure we are left with at least a constant fraction of the original utility. We give the proof in the Appendix.

▶ **Lemma 2.** *Given a packing $p : I' \to [T]$ of items $I' \subseteq I$, let $p' : I'' \to [T] \setminus E$ be the packing obtained by applying Algorithm 2. Then,*

$$\sum_{i \in I''} u_{ip'(i)} \geq \frac{\pi_3}{2} \sum_{i \in I'} u_{ip(i)}.$$

Repacking. We now show that for any $\lambda \in (0, \frac{1}{5})$ and suitable values of $0 < \pi_1 < \pi_2 < 1$, and $\pi_3 \in [2\lambda, 1)$, we can always pack the items in J' into the empty bins obtained from the above eviction process. We prove a slightly more general result.

▶ **Theorem 3** (Repacking Theorem)**.** *In a laminar instance of χ-AGAP, let $\hat{J}$ be a set of groups and $\hat{T} \subseteq [T]$ a subset of bins. Suppose that for parameters $\{\alpha, \beta, \gamma\} \in (0,1)$ the following conditions are satisfied:*
(a) *For any $j \in \hat{J}$, $|\hat{I}_j| \leq \gamma |\chi_j|$*
(b) *For any $\chi \in \mathcal{L}$, there are at least $\alpha|\chi|$ bins in $\hat{T} \cap \chi$*
(c) *For any $\chi \in \mathcal{L}$, $\sum_{j \in \hat{J} : \chi_j \subseteq \chi} a_j \leq \beta |\chi|$.*
Then we can feasibly pack all the items $\hat{J}$ into $\hat{T}$ if $\gamma + 2\beta \leq \alpha$.

Proof. Let $S = \cup_{j \in \hat{J}} \hat{I}_j$ be the set of all items in the groups $\hat{J}$. In the course of our algorithm, we label bins with one of the possible three colors: *white, gray* and *black*. Our algorithm works in a *bottom-up* fashion and marks an interval χ *done* when it has successfully packed all the groups $j \in \hat{J}$ such that $\chi_j \subseteq \chi$. Initially all the bins are marked *white*. A bin t is marked *gray* when we pack an item into t, and black when we decide to add no more items to this bin. Consider an interval χ that has not been marked *done*, but every interval $\chi' \subset \chi$ is marked *done*. Let j be a group with $\chi_j = \chi$ that has not been packed completely. Pick an unpacked item $i \in I_j$. Let *avail*$(j) \subseteq \chi \cap \hat{T}$ be the set of bins that are not packed with items in group j. Algorithm 3 describes the formal procedure to pack item i.

We show that Algorithm 3 never reports **fail** and therefore feasibly packs all the items in S. Assume towards contradiction that the algorithm reports **fail** while packing an item i. Define a bin $t \in \chi \cap \hat{T}$ as *bad* if t is colored gray or some other item $i' \in \hat{I}_j$ has been packed in t. We first show that the following invariant holds, as long as no item in a group j^+ such that $\chi \subset \chi_{j^+}$ has been packed: the number of bad bins while processing group j is at most $\gamma|\chi|$. Assuming that the claim holds for each child interval of χ, namely, $\{\chi_1, \chi_2 \ldots, \chi_s\}$, before any group with time window χ is processed, we have the number of bad bins = number of

gray bins is at most $\sum_{l\in[s]}\gamma|\chi_l| \le \gamma|\chi|$. Now, consider the iteration where we pack some item $i \in \hat{I}_j$ into a bin t. If t is a gray bin, then the number of bad bins cannot increase. On the other hand, suppose t was white before packing i. If there are no gray bins in χ, then the number of bad bins is at most $|\hat{I}_j| \le \gamma|\chi|$. Suppose there exist some gray bins, and consider those bins t' such that group j has no item packed into bin t'. If there are no such bins, then again the number of bad bins is at most $|\hat{I}_j| \le \gamma|\chi|$. Otherwise, we must have considered one such gray bin t' and failed to pack i into it. By the virtue of the algorithm, we must have colored both t and t' black. Thus, the number of bad bins would not increase, and our claim holds. Now, since we pair the black bins $t \leftrightarrow t'$ only if $\sum_{i\in S_t} s_i + \sum_{i'\in S'_t} s_{i'} > 1$, the total number of black bins $< 2\beta|\chi|$. Hence, the total number of bins that are black or bad $< (\gamma + 2\beta)|\chi|$. Setting $\alpha \ge (\gamma + 2\beta)$, there should be at least one bin t^* that is neither black nor bad. But in this case, we could have packed i into t^* – a contradiction to the assumption the algorithm reports a **fail**. ◀

Putting it all together, we obtain the following main result.

▶ **Theorem 4.** *For any slackness parameter $\lambda \in (0, \frac{1}{5})$, there exists a polynomial time $\Omega(1)$-approximation algorithm for laminar χ-AGAP.*

Proof. Suppose we are given a laminar instance $\mathcal{I} = (I, J, T, \mathcal{L})$ of χ-AGAP with optimal profit OPT. We apply Lemma 1 to obtain a subset of groups J' and a *partial* packing p of a subset of items in J', denoted by I', such that

1. For any $\chi \in \mathcal{L}$, $\sum_{j\in J':\chi_j\subseteq\chi} a_j \le (\pi_2 + \lambda)|\chi|$
2. $\mathbb{E}\left[\sum_{i\in I'} u_{ip(i)}\right] \ge \frac{\pi_1}{2}\left(1 - \left(\frac{\pi_1}{\pi_2} + \pi_1\right)\right) OPT$.

Further, such a packing respects the capacity constraints and also ensures that no two items of the same group are packed in the same bin. Choosing $\pi_3 \ge 2\lambda$, we apply Algorithm 2 to compute a set of evicted bins E such that $|E \cap \chi| \ge \lceil(1 - \pi_3)|\chi|\rceil$, for any $\chi \in \mathcal{L}$. By Lemma 2 and property 2. above, the new packing $p' : I'' \to [T] \setminus E$ satisfies

$$\mathbb{E}\left[\sum_{i\in I''} u_{ip'(i)}\right] \ge \frac{\pi_3}{2}\mathbb{E}\left[\sum_{i\in I'} u_{ip(i)}\right] \ge \frac{\pi_1\pi_3}{4}\left(1 - \left(\frac{\pi_1}{\pi_2} + \pi_1\right)\right) OPT.$$

Let $\hat{I}_j = I_j \setminus I''$ be the unpacked items in group j, for all $j \in J'$, and let $\hat{J} = \cup_{j\in J'}\hat{I}_j$. Also, we can write $\pi_3 = 2\lambda + \pi'_3$, for $\pi'_3 \ge 0$. We show below that, for any $\lambda \in (0, \frac{1}{5})$, there exist positive constants π_1, π_2 and non-negative constant π'_3, such that (a) $\lambda + 2(\pi_2 + \lambda) \le 1 - \pi_3$ and (b) $\frac{\pi_1}{\pi_2} + \pi_1 < 1$. With $\lambda \in (0, \frac{1}{5})$, we can select a positive constant value for π_2 and a non-negative value π'_3, such that $5\lambda \le 1 - \pi'_3 - 2\pi_2$ (this is possible since $1 - 5\lambda > 0$). It follows that $5\lambda \le 1 - (\pi_3 - 2\lambda) - 2\pi_2$, and (a) is satisfied. Now, setting $\pi_1 < \frac{1}{1+1/\pi_2}$, we ensure that (b) holds. To complete the proof, we show that the above implies that (i) all of the items in $\hat{J}$ can be packed into the empty bins in E, and (ii) the total expected utility is a constant fraction of the optimum. To show (i), we apply Theorem 3, setting $\hat{T} = E$, $\alpha = 1 - \pi_3$, $\beta = \pi_2 + \lambda$ and $\gamma = \lambda$. Then, from (a) we have that $\gamma + 2\beta \le \alpha$. Thus, all of the conditions of Theorem 3 hold and we can pack all the items $\hat{J}$. Now, (ii) follows from the fact that $\frac{\pi_1\pi_3}{4}\left(1 - \left(\frac{\pi_1}{\pi_2} + \pi_1\right)\right)$ is a positive constant. Thus, we obtain a constant approximation for any $\lambda \in (0, \frac{1}{5})$. ◀

4 The General Case

In this section we extend our approach for laminar instances to the general case. As a first step, we transform the instance to a family of *laminar* intervals. Consider a general χ-AGAP

Algorithm 4 Transformation of a general family of intervals into a Laminar family.

Input: Job set J and $\mathcal{W} = \{\chi_j : j \in J\}$

Output: Laminar family of intervals $\mathcal{L}$ and a mapping $\mathfrak{L} : \mathcal{W} \to \mathcal{L}$

1: Construct a binary tree with the interval $[T]$ as the root. Each node in the tree corresponds to an interval $\chi = [l, r] \subseteq [T]$.

2: If $r - l > 1$, then the node $[l, r]$ has two children corresponding to intervals $[l, \lfloor \frac{l+r}{2} \rfloor]$ and $[\lfloor \frac{l+r}{2} \rfloor + 1, r]$. We define $\mathcal{L}$ as the union of intervals corresponding to the nodes in the tree.

3: For each $\chi \in \mathcal{W}$, let χ' be the largest interval in $\mathcal{L}$ contained in χ, breaking ties by picking the *rightmost* interval, then $\mathfrak{L}(\chi) = \chi'$.

instance. Let $\mathcal{W}$ denote the set of all time-windows for groups in J, i.e., $\mathcal{W} = \{\chi_j : j \in J\}$. We now construct a laminar family of intervals $\mathcal{L}$ and a mapping $\mathfrak{L} : \mathcal{W} \to \mathcal{L}$. Recall that $T = \max_{j \in J} d_j$.

▶ **Construction 5.** *The construction is formally described in Algorithm 4.*

It is natural to consider transforming a general instance to a laminar instance, using Algorithm 4, and then applying the LP rounding procedure (of Section 3) to the laminar instance. However, this can lead to a solution that is far from the optimal. Indeed, as item utilities depend on the bins, the configurations for the general instance can differ significantly from those of the laminar instance. Specifically, it may be the case that for some group $j \in J$, item $i \in I_j$ has high utility when packed in a bin $t \in \chi_j$, but $t \notin \mathfrak{L}(\chi_j)$. We overcome this issue by defining the configurations on the *original* intervals $\mathcal{W}$, while setting the knapsack constraints (in Step. 5 of Algorithm 1) on the intervals $\mathcal{L}$.

We give the details of the algorithm in the Appendix.

References

1 Ron Adany, Moran Feldman, Elad Haramaty, Rohit Khandekar, Baruch Schieber, Roy Schwartz, Hadas Shachnai, and Tami Tamir. All-or-nothing generalized assignment with application to scheduling advertising campaigns. In *IPCO*, pages 13–24. Springer, 2013.

2 Micah Adler, Phillip B Gibbons, and Yossi Matias. Scheduling space-sharing for internet advertising. *Journal of Scheduling*, 5(2):103–119, 2002.

3 V. Balachandran. An integer generalized transportation model for optimal job assignment in computer networks. *Operations Research*, 24(4):742–759, 1976.

4 Nikhil Bansal, Nitish Korula, Viswanath Nagarajan, and Aravind Srinivasan. Solving packing integer programs via randomized rounding with alterations. *Theory of Computing*, 8(1):533–565, 2012.

5 Marco Bender, Clemens Thielen, and Stephan Westphal. Packing items into several bins facilitates approximating the separable assignment problem. *Information Processing Letters*, 115(6-8):570–575, 2015.

6 Niv Buchbinder and Moran Feldman. Deterministic algorithms for submodular maximization problems. In *SODA*, pages 392–403, 2016.

7 C. Chekuri and S. Khanna. A PTAS for the multiple knapsack problem. *SIAM J. on Computing*, 35(3):713–728, 2006.

8 Lin Chen and Guochuan Zhang. Packing groups of items into multiple knapsacks. In *STACS*, pages 28:1–28:13, 2016.

9 NM Mosharaf Kabir Chowdhury, Muntasir Raihan Rahman, and Raouf Boutaba. Virtual network embedding with coordinated node and link mapping. In *INFOCOM*, pages 783–791. IEEE, 2009.

10 Robert G Cromley and Dean M Hanink. Coupling land use allocation models with raster GIS. *Journal of Geographical Systems*, 1(2):137–153, 1999.

11 Marek Cygan, Fabrizio Grandoni, and Monaldo Mastrolilli. How to sell hyperedges: The hypermatching assignment problem. In *SODA*, pages 342–351. SIAM, 2013.

12 Milind Dawande, Subodha Kumar, and Chelliah Sriskandarajah. Performance bounds of algorithms for scheduling advertisements on a web page. *Journal of Scheduling*, 6(4):373–394, 2003.

13 Milind Dawande, Subodha Kumar, and Chelliah Sriskandarajah. Scheduling web advertisements: a note on the minspace problem. *Journal of Scheduling*, 8(1):97–106, 2005.

14 Uriel Feige and Jan Vondrák. Approximation algorithms for allocation problems: Improving the factor of 1-1/e. In *FOCS*, pages 667–676, 2006.

15 Lisa Fleischer, Michel X Goemans, Vahab S Mirrokni, and Maxim Sviridenko. Tight approximation algorithms for maximum separable assignment problems. *Mathematics of Operations Research*, 36(3):416–431, 2011.

16 Ari Freund and Joseph Naor. Approximating the advertisement placement problem. *Journal of Scheduling*, 7(5):365–374, 2004.

17 Francesca Guerriero, Giovanna Miglionico, and Filomena Olivito. Managing tv commercials inventory in the italian advertising market. *International Journal of Production Research*, 54(18):5499–5521, 2016.

18 Arshia Kaul, Sugandha Aggarwal, Anshu Gupta, Niraj Dayama, Mohan Krishnamoorthy, and PC Jha. Optimal advertising on a two-dimensional web banner. *International Journal of System Assurance Engineering and Management*, pages 1–6, 2017.

19 Subodha Kumar, Milind Dawande, and Vijay Mookerjee. Optimal scheduling and placement of internet banner advertisements. *IEEE Transactions on Knowledge and Data Engineering*, 19(11), 2007.

20 Amin Ghalami Osgouei, Amir Khorsandi Koohanestani, Hossein Saidi, and Ali Fanian. Online assignment of non-SDN virtual network nodes to a physical SDN. *Computer Networks*, 129:105–116, 2017.

21 Mark J Panaggio, Pak-Wing Fok, Ghan S Bhatt, Simon Burhoe, Michael Capps, Christina J Edholm, Fadoua El Moustaid, Tegan Emerson, Star-Lena Estock, Nathan Gold, Ryan Halabi, Madelyn Houser, Peter R Kramer, Hsuan-Wei Lee, Qingxia Li, Weiqiang Li, Dan Lu, Yuzhou Qian, Louis F Rossi, Deborah Shutt, Vicky Chuqiao Yang, and Yingxiang Zhou. Prediction and optimal scheduling of advertisements in linear television. *arXiv preprint arXiv:1608.07305*, 2016.

22 Shinjini Pandey, Goutam Dutta, and Harit Joshi. Survey on revenue management in media and broadcasting. *Interfaces*, 47(3):195–213, 2017.

23 Adil Razzaq, Peter Sjödin, and Markus Hidell. Minimizing bottleneck nodes of a substrate in virtual network embedding. In *NOF*, pages 35–40, 2011.

24 Robert Ricci, Chris Alfeld, and Jay Lepreau. A solver for the network testbed mapping problem. *ACM SIGCOMM Computer Communication Review*, 33(2):65–81, 2003.

25 Kanthi K. Sarpatwar, Baruch Schieber, and Hadas Shachnai. Brief announcement: Approximation algorithms for preemptive resource allocation. *To appear in SPAA*, 2018.

26 SintecMedia - On Air. http://www.sintecmedia.com/OnAir.html, 2013.

27 Jan Vondrák. Optimal approximation for the submodular welfare problem in the value oracle model. In *STOC*, pages 67–74, 2008.

28 Minlan Yu, Yung Yi, Jennifer Rexford, and Mung Chiang. Rethinking virtual network embedding: substrate support for path splitting and migration. *Computer Communication Review*, 38(2):17–29, 2008.

A Some Proofs

Proof of Lemma 1 (parts (ii) and (iii)):

(ii) We show (2) using induction on the intervals χ.
Base Step. Let χ be an interval containing no other interval in $\mathcal{L}$, i.e., there is no $\chi' \in \mathcal{L}$ such that $\chi' \subset \chi$. Let j be the last group with time window $\chi_j = \chi$, in the ordering $\mathcal{O}$, that is successfully added to J'; if there is no such group, we are done. Consider the iteration (in Algorithm 1, Step 3.) in which j was considered. In this iteration, we must have that $\sum_{j' \in J' : \chi_{j'} \subseteq \chi} a_{j'} \leq \pi_2 |\chi|$ – otherwise, j should have been discarded. Thus, upon adding j to J', we have

$$\sum_{j' \in J' : \chi_{j'} \subseteq \chi} a_{j'} \leq \pi_2 |\chi| + a_j \leq \pi_2 |\chi| + \lambda |\chi_j| = (\pi_2 + \lambda)|\chi|.$$

The second inequality holds since $a_j = \sum_{i \in I_j} s_i \leq |I_j| \leq \lambda |X_j|$.
Inductive Step. Let χ be an interval with child intervals $\chi_1, \chi_2, \ldots, \chi_s$ such that the claim holds for each χ_k, $k \in [s]$. First, suppose there is no group $j \in J'$ with $\chi_j = \chi$. In this case, we have

$$\sum_{j' \in J' : \chi_{j'} \subseteq \chi} a_{j'} = \sum_{k \in [s]} \sum_{j' \in J' : \chi_{j'} \subseteq \chi_k} a_{j'} \leq \sum_{k \in [s]} (\pi_2 + \lambda)|\chi_k| \leq (\pi_2 + \lambda)|\chi|.$$

Now, consider the case where there is a group $j \in J'$ with $\chi_j = \chi$ and w.l.o.g. let j be the last such group. As in the base case, in the iteration where j was added to J', we have $\sum_{j' \in J' : \chi_{j'} \subseteq \chi} a_{j'} \leq \pi_2 |\chi|$. Thus, after adding j to J', we have $\sum_{j' \in J' : \chi_{j'} \subseteq \chi} a_{j'} \leq (\pi_2 + \lambda)|\chi|$.
(iii) As shown in part (i) of the proof, for any bin t, we either keep the last item i_t, or the rest of the items, $S_t \setminus i_t$, and both respect the capacity constraint corresponding to t. ◀

Proof of Lemma 2:

Let $U_t = \sum_{i \in I' : p(i) = t} u_{it}$ denote the total utility of items in I' packed into t under the packing p. Using induction, we show that: in the iteration where an interval χ is marked processed, the total utility of items packed, under p', into χ is at least $\sum_{t \in \chi} \frac{\pi_3}{2} U_t$. We note that this claim may not hold for χ in subsequent iterations, when we process a parent interval $\chi^+ \supset \chi$. However, at the end of all iterations, this claim would ensure that under p' we are left with at least a constant fraction of the total profit under p, i.e., $\sum_{t \in [T]} \sum_{i \in I'' : p'(i) = t} u_{it} \geq \sum_{t \in [T]} \frac{\pi_3}{2} U_t$.
Base Case. Let $\chi \in \mathcal{L}$ be a leaf interval i.e., there is no $\chi' \in \mathcal{L}$ such that $\chi' \subset \chi$. Note that we process χ before any of its parent intervals $\chi^+ \supset \chi$. When χ is marked as *processed*, we would have evicted at most $\lceil (1 - \pi_3)|\chi| \rceil$ least profitable bins. Further,

$$\begin{aligned} \lceil (1 - \pi_3)|\chi| \rceil &\leq (1 - \pi_3)|\chi| + 1 \leq (1 - \pi_3)|\chi| + \frac{\pi_3}{2\lambda} \\ &\leq (1 - \pi_3)|\chi| + \frac{\pi_3}{2}|\chi| = (1 - \frac{\pi_3}{2})|\chi|. \end{aligned} \tag{10}$$

The second inequality follows from the choice of $\pi_3 \geq 2\lambda$. For the last inequality, we note that there exists a group $j \in J$ with $\chi_j = \chi$ and therefore $1 \leq I_j \leq \lambda |\chi|$. Thus, in this case, at least $\frac{\pi_3}{2}|\chi|$ of the most profitable bins are not evicted, and the claim holds.
Inductive Step. Suppose χ is a non-leaf interval with children $\chi_l, l \in [s]$. We consider the iteration in which χ was marked *processed*. First, we observe that, at this iteration, the total number of bins that are not evicted in χ is at least $\frac{\pi_3}{2}|\chi|$. To see this, consider the following two cases:

(i) If some bins are evicted in the iteration where χ is being processed (Step 3. of Algorithm 2), then the number of evicted bins in χ is exactly $\lceil(1-\pi_3)|\chi|\rceil \leq (1-\frac{\pi_3}{2})|\chi|$ (by Equation (10).

(ii) If no new bin is evicted in the iteration then, by the algorithm and Equation (10), the total number of evicted bins in χ is bounded by $\sum_{l\in[s]}\lceil(1-\pi_3)|\chi_l|\rceil \leq \sum_{l\in[s]}(1-\frac{\pi_3}{2})|\chi_l| \leq (1-\frac{\pi_3}{2})|\chi|$.

We use a charging argument to prove our original claim, i.e.,

$$\sum_{t\in\chi}\sum_{\{i\in I'':p'(i)=t\}} u_{it} \geq \frac{\pi_3}{2}\sum_{t\in\chi} U_t.$$

Every bin (evicted or otherwise) $t \in \chi$ is charged to a bin $t' \in \chi \setminus E$ that is not evicted, such that

1. the total profit of items packed in t' is greater than those packed in t, i.e., $U_{t'} \geq U_t$.
2. no bin t' is charged with more than $\frac{2}{\pi_3}$ bins.

This would clearly imply that the total utility of non-evicted bins is at least $\frac{\pi_3}{2}$ fraction of $\sum_{t\in\chi} U_t$. The charging is again done in a bottom-up fashion. For a leaf interval: a non-evicted bin is charged to itself; evicted bins are arbitrarily distributed among the non-evicted bins in such a way that no bin is charged with more than $\frac{2}{\pi_3}$ bins. This is possible because there are at least $\frac{\pi_3}{2}|\chi|$ bins that are not evicted.

Suppose that all the bins in each child interval, $\chi_l : l \in [s]$, of χ has been charged. When considering χ, some new bins may be evicted. Consider any newly evicted bin t that in turn might have been charged with a certain subset of evicted bins. We observe that all the bins in χ that are not evicted must have profit at least as much as t and are therefore more profitable than the evicted bins charged to t. Thus, we can arbitrarily assign these evicted bins (bins charged to t, including itself) to the non-evicted bins, as long as the number of bins charged to any bin is at most $\frac{2}{\pi_3}$. This is possible because we have at most $(1-\frac{\pi_3}{2})|\chi|$ evicted bins in χ at this iteration (i.e., when χ was marked processed). The claim follows by induction.

B Approximating General χ-AGAP instances

We describe below the extension of our algorithm for laminar instances to handle general χ-AGAP instances.

Phase 1. The LP remains the same. Note that the configurations are defined on the set of intervals $\mathcal{W}$

$$\text{Maximize}\left\{\sum_{M\in\mathcal{C}} u_M x_M : \sum_{M\in\mathcal{C}_t} x_M s_{M^{-1}(t)} \leq 1\,(\forall t\in[T]),\ \sum_{M\in\mathcal{C}_j} x_M \leq 1\,(\forall j\in J),\ x_M \geq 0\right\}$$

Rounding Algorithm. As in the laminar case, fix suitable constants $\pi_1, \pi_2 \in (0,1)$ and solve the configuration LP to obtain an optimal fractional solution $(x^*_M : M \in \mathcal{C})$. Now, using Algorithm 4, transform $\mathcal{W}$ into a family of laminar intervals $\mathcal{L}$ and obtain a mapping $\mathfrak{L} : \mathcal{W} \to \mathcal{L}$ of intervals in $\mathcal{W}$ to those in $\mathcal{L}$. Next, we associate each interval $\chi \in \mathcal{L}$ with a knapsack capacity of $\pi_2|\chi|$. Further, the ordering of groups $\mathcal{O}$ is based on $\mathfrak{L}$ (and $\mathcal{L}$). That is, if j and ℓ are two groups such that $\mathfrak{L}(\chi_j) \subseteq \mathfrak{L}(\chi_\ell)$, then j appears before ℓ in $\mathcal{O}$.

While considering group j, we check if the knapsack capacity corresponding to $\mathfrak{L}(\chi_j)$ has been violated. Formally, let J' be the set of groups chosen before j; then, we check if

Algorithm 5 Rounding the configuration LP.

Input: A χ-AGAP instance: $(I, J, T, \mathcal{W})$
Output: $J' \subseteq J$, $I' \subseteq \bigcup_{j \in J'} I_j$ and $p : I' \to [T]$

Initialize $J' \leftarrow \emptyset$ and $I' \leftarrow \emptyset$.
Solve the configuration LP and obtain an optimal fractional solution $(x^*_M : M \in \mathcal{C})$
Compute an ordering $\mathcal{O}$ of groups in J as follows: for any $j, \ell \in J$, if $\mathfrak{L}(\chi_j) \subseteq \mathfrak{L}(\chi_\ell)$ then j appears before j' in $\mathcal{O}$
for all j in accordance with the ordering $\mathcal{O}$ **do**
    Let $X_j = \sum_{M \in \mathcal{C}_j} x^*_M$.
    $\sum_{j' \in J' : \mathfrak{L}(\chi_{j'}) \subseteq \mathfrak{L}(\chi_j)} a_{j'} \leq \pi_2 |\mathfrak{L}(\chi_j)|$, set $J' \leftarrow J' \cup \{j\}$ with probability $\pi_1 X_j$.
    If $j \in J'$, let $\mathcal{C}^+_j$ denote the configurations $M \in \mathcal{C}_j$ such that $x^*_M > 0$.
    Choose exactly one configuration, $M_j \in \mathcal{C}^+_j$ with probability $\frac{x^*_M}{X_j}$ for configuration M.
    For all $(i, t) \in M_j$, if $\sum_{i' \in I' : p(i') = t} s_{i'} \leq 1$, set $I' \leftarrow I' \cup \{i\}$ and $p(i) = t$.
end for
for all $t \in [T]$ **do**
    if $\sum_{i' \in I' : p(i') = t} s_{i'} > 1$ **then** ▷ *handle violated bins*
        Let i be the last item with $p(i) = t$. If $u_{it} \geq \sum_{i' : p(i') = t} u_{i't}$: set $p(i') = \emptyset$ and $I' \leftarrow I' \setminus \{i'\}$ $\forall i' \neq i$; Otherwise, set $p(i) = \emptyset$ and $I' \leftarrow I' \setminus \{i\}$.
    end if
end for

$\sum_{j' \in J' : \mathfrak{L}(\chi_{j'}) \subseteq \mathfrak{L}(\chi_j)} a_{j'} \leq \pi_2 |\mathfrak{L}(\chi_j)|$. If this constraint holds, we add j into J' with probability $\pi_1 \sum_{M \in \mathcal{C}_j} x^*_M = \pi_1 X_j$, where $X_j = \sum_{M \in \mathcal{C}_j} x^*_M$; otherwise, we discard j. If j has been selected, i.e., $j \in J'$, then choosing a configuration and then picking a subset of items in I_j that can be packed is exactly the same as in the laminar case. Our rounding scheme is formally described in Algorithm 5.

The proof of the following lemma is similar to the proof of Lemma 1 (details omitted).

▶ **Lemma 6.** *For suitable constants π_1, π_2, λ, assuming that for any group $j \in J$, $|I_j| \leq \lambda |\chi_j|$, the following hold for Algorithm 5:*

(i) *The expected utility of the items in I' is at least a constant fraction of the optimal LP value, i.e.,*

$$\mathbb{E}\left[\sum_{i \in I'} u_{ip(i)}\right] \geq \frac{\pi_1}{2}\left(1 - \left(\frac{4\pi_1}{\pi_2} + \pi_1\right)\right)\left[\sum_{M \in \mathcal{C}} u_M x^*_M\right]$$

(ii) *For any interval $\chi \in \mathcal{L}$, the total size of groups chosen in χ is at most a constant fraction of $|\chi|$, i.e.,*

$$\sum_{j \in J' : \mathfrak{L}(\chi_j) \subseteq \chi} a_j \leq (\pi_2 + 4\lambda)|\chi|$$

(iii) *For any bin $t \in T$, the total size of items packed into t is within its capacity, i.e.,*

$$\sum_{i \in I' : p(i) = t} s_i \leq 1$$

Phase 2. We now apply the eviction and repacking phase on the transformed instance, i.e., considering the time windows $\chi \in \mathcal{L}$. We summarize in the next theorem.

We use in the analysis the next result (due to [25]). We include a proof for completeness.

▶ **Lemma 7.** *In Algorithm 4, the following properties hold:*

1. *For any $j \in J$, $|\chi_j| \leq 4|\mathfrak{L}(\chi_j)|$*
2. *For $\chi \in \mathcal{L}$, let $\tilde{\chi} = \{t \in \chi_j : j \in J, \mathfrak{L}(\chi_j) = \chi\}$, i.e., the union of all time-windows in $\mathcal{W}$ that are mapped to χ. Then, $|\tilde{\chi}| \leq 4|\chi|$.*

Proof. To prove the first property, it suffices to show that χ_j cannot completely contain 3 consecutive intervals in $\mathcal{L}$ that are at the same level as $\mathfrak{L}(\chi_j)$. Indeed, this would imply that χ_j cannot intersect more than 4 consecutive intervals, and therefore $|\chi_j| \leq 4|\mathfrak{L}(\chi_j)|$. Now, suppose χ_j contains at least 3 such consecutive intervals. Then, by the virtue of our algorithm, $\mathfrak{L}(\chi_j)$ is the rightmost interval. Let $\hat{\chi}$ be the parent of $\mathfrak{L}(\chi_j)$. Two cases arise:

Case 1: $\mathfrak{L}(\chi_j)$ is a left child of $\hat{\chi}$. Consider the two other consecutive intervals at the same level as $\mathfrak{L}(\chi_j)$ that are contained in χ_j. Observe that these two intervals are siblings; therefore, their parent (which is also in $\mathcal{L}$) is also contained in χ_j. This is a contradiction to the assumption that $\mathfrak{L}(\chi_j)$ is the largest interval in $\mathcal{L}$ contained in χ_j.

Case 2: $\mathfrak{L}(\chi_j)$ is a right child of $\hat{\chi}$. We observe that the sibling of $\mathfrak{L}(\chi_j)$ must also be contained in χ_j, implying that $\hat{\chi}$ is contained in χ_j, a contradiction.

We now prove the second property. For any $\chi = [s, d] \in \mathcal{L}$, let $\chi_l = [s_l, d_l] \in \mathcal{W}$ (resp. $\chi_r = [s_r, d_r]$) be the leftmost (resp. rightmost) interval in $\mathcal{W}$ such that $\mathfrak{L}(\chi_l) = \chi$ (resp. $\mathfrak{L}(\chi_r) = \chi$); then, $\tilde{\chi} = [s_l, d_r]$. Consider the intervals $\chi_1 = [s_l, s]$ and $\chi_2 = [d, d_r]$. As argued above, χ_l cannot contain 3 consecutive intervals in $\mathcal{L}$ at the same level as χ Thus, $|\chi_1| < 2|\chi|$.

Also, $|\chi_2| < |\chi|$; otherwise, there is an interval to the right of χ of the same size that can be mapped to χ_r. Thus, $|\chi_1| + |\chi_2| < 3|\chi|$. Now, the claim follows by observing that $|\tilde{\chi}| = |\chi_1| + |\chi| + |\chi_2| \leq 4|\chi|$. ◀

▶ **Theorem 8.** *For any slackness parameter $\lambda \in (0, \frac{1}{20})$, there exists a polynomial time $\Omega(1)$-approximation algorithm for general χ-AGAP instances.*

Proof. Suppose that the given instance $\mathcal{I} = (I, J, T, \mathcal{W})$ has an optimal profit OPT. We apply Lemma 6 to obtain a subset of groups J' and a *partial* packing p of a subset of items in J', denoted by I', such that:

1. For any $\chi \in \mathcal{L}$, $\sum_{j \in J' : \mathfrak{L}(\chi_j) \subseteq \mathfrak{L}(\chi)} a_j \leq (\pi_2 + 4\lambda)|\chi|$
2. $\mathbb{E}\left[\sum_{i \in I'} u_{ip(i)}\right] \geq \frac{\pi_1}{2}\left(1 - \left(\frac{4\pi_1}{\pi_2} + \pi_1\right)\right) OPT$.

Further, such a packing respects the capacity constraints and also ensures that no two items of the same group are packed in the same bin. Choosing a suitable $\pi_3 \geq 8\lambda$, we apply Algorithm 2 on the intervals $\mathcal{L}$ to compute a set of evicted bins E such that $|E \cap \chi| \geq (1 - \pi_3)|\chi|$, for any $\chi \in \mathcal{L}$. By Lemma 2 and property 2. above, the new packing $p' : I'' \to [T] \setminus E$ satisfies

$$\mathbb{E}\left[\sum_{i \in I''} u_{ip'(i)}\right] \geq \frac{\pi_3}{2}\mathbb{E}\left[\sum_{i \in I'} u_{ip(i)}\right] \geq \frac{\pi_1\pi_3}{4}\left(1 - \left(\frac{4\pi_1}{\pi_2} + \pi_1\right)\right) OPT.$$

Let $\hat{I}_j = I_j \setminus I''$ be the unassigned items in group j, for all $j \in J'$, and let $\hat{J} = \cup_{j \in J'} \hat{I}_j$. Also, $\pi_3 = 8\lambda + \pi_3'$, for some $\pi_3' \geq 0$. We show below that, for any $\lambda \in (0, \frac{1}{20})$, there exist constants π_1, π_2, π_3', such that (a) $4\lambda + 2(\pi_2 + 4\lambda) \leq 1 - \pi_3$ and (b) $\frac{4\pi_1}{\pi_2} + \pi_1 < 1$. Let $\lambda \in (0, \frac{1}{20})$, and select π_2, π_3', such that $20\lambda \leq 1 - \pi_3' - 2\pi_2$. Then, $12\lambda + 2(\pi_2 + 4\lambda) \leq 1 - \pi_3$, and (a) is satisfied. Now, setting $\pi_1 < \frac{1}{1 + \frac{4}{\pi_2}}$, we have that (b) holds. To complete the proof, we show that the above implies that (i) all of the items in $\hat{J}$ can be packed into the empty bins in E, and (ii) the total expected utility is a constant fraction of the optimum. To show (i),

we apply Theorem 3, setting $\hat{T} = E$, $\alpha = 1 - \pi_3$, $\beta = \pi_2 + 4\lambda$ and $\gamma = 4\lambda$. Then, from (a) we have that $\gamma + 2\beta \leq \alpha$. Also, we note that, by Lemma 7, for all $j \in \hat{J}$ such that $\chi = \mathfrak{L}(\chi_j)$, it holds that $|\hat{I}_j| \leq |I_j| \leq \lambda|\chi_j| \leq 4\lambda|\mathfrak{L}(\chi_j)| = \gamma|\chi|$ Thus, all of the conditions of Theorem 3 hold. Now, (ii) follows from (b). Hence, we obtain a constant approximation for any $\lambda \in (0, \frac{1}{20})$. ◀

On Geodesically Convex Formulations for the Brascamp-Lieb Constant

Suvrit Sra
Massachusetts Institute of Technology (MIT), Cambridge, MA, USA

Nisheeth K. Vishnoi
École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland

Ozan Yıldız
École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland

Abstract

We consider two non-convex formulations for computing the optimal constant in the Brascamp-Lieb inequality corresponding to a given datum and show that they are geodesically log-concave on the manifold of positive definite matrices endowed with the Riemannian metric corresponding to the Hessian of the log-determinant function. The first formulation is present in the work of Lieb [15] and the second is new and inspired by the work of Bennett *et al.* [5]. Recent work of Garg *et al.* [12] also implies a geodesically log-concave formulation of the Brascamp-Lieb constant through a reduction to the operator scaling problem. However, the dimension of the arising optimization problem in their reduction depends exponentially on the number of bits needed to describe the Brascamp-Lieb datum. The formulations presented here have dimensions that are polynomial in the bit complexity of the input datum.

2012 ACM Subject Classification Theory of computation → Nonconvex optimization, Theory of computation → Convex optimization

Keywords and phrases Geodesic convexity, positive definite cone, geodesics, Brascamp-Lieb constant

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.25

Acknowledgements We thank Ankit Garg for useful discussions on operator scaling and its connections to Brascamp-Lieb constant.

1 Introduction

The Brascamp-Lieb Inequality. Brascamp and Lieb [7] presented a class of inequalities that generalize many well-known inequalities and, as a consequence, have played an important role in various mathematical disciplines. Formally, they presented the following class of inequalities where each inequality is described by a "datum", referred to as the Brascamp-Lieb datum.

▶ **Definition 1** (The Brascamp-Lieb Inequality, Datum, Constant)**.** Let n, m, and $(n_j)_{j\in[m]}$ be positive integers and $p := (p_j)_{j\in[m]}$ be non-negative real numbers. Let $B := (B_j)_{j\in[m]}$ be an m-tuple of linear transformations where B_j is a surjective linear transformation from $\mathbb{R}^n$ to $\mathbb{R}^{n_j}$. The corresponding Brascamp-Lieb *datum* is denoted by (B, p). The Brascamp-Lieb inequality states that for each Brascamp-Lieb datum (B, p) there exists a constant $C(B, p)$ (not necessarily finite) such that for any selection of real-valued, non-negative, Lebesgue

measurable functions f_j where $f_j : \mathbb{R}^{n_j} \to \mathbb{R}$,

$$\int_{x\in\mathbb{R}^n} \left(\prod_{j\in[m]} f_j(B_j x)^{p_j} \right) dx \leq C(B,p) \prod_{j\in[m]} \left(\int_{x\in\mathbb{R}^{n_j}} f_j(x)dx \right)^{p_j}. \tag{1}$$

The smallest constant that satisfies (1) for any choice of $f := (f_j)_{j\in[m]}$ satisfying the properties mentioned above is called the Brascamp-Lieb *constant* and is denoted by $\mathrm{BL}(B,p)$. A Brascamp-Lieb datum (B,p) is called *feasible* if $\mathrm{BL}(B,p)$ is finite, otherwise, it is called *infeasible*. For a given m-tuple B, the set of real vectors p such that (B,p) is feasible is denoted by P_B.

Applications of the Brascamp-Lieb inequality extend beyond functional analysis and appear in convex geometry [3], information theory [8],[16],[17], machine learning [14], and theoretical computer science [11, 10].

Mathematical Aspects of the Brascamp-Lieb Inequality. A Brascamp-Lieb inequality is non-trivial only when (B,p) is a feasible Brascamp-Lieb datum. Therefore, it is of interest to characterize feasible Brascamp-Lieb data and compute the corresponding Brascamp-Lieb constant. Lieb [15] showed that one needs to consider only Gaussian functions as inputs for (1). This result suggests the following characterization of the Brascamp-Lieb constant as an optimization problem. For a positive integer k, let $\mathbb{S}_+^k$ be the space of real-valued, symmetric, positive semi-definite (PSD) matrices of dimension $k \times k$.

▶ **Theorem 2** (Gaussian maximizers [15]). *Let (B,p) be a Brascamp-Lieb datum with $B_j \in \mathbb{R}^{n_j\times n}$ for each $j \in [m]$. Let $A := (A_j)_{j\in[m]}$ with $A_j \in \mathbb{S}_+^{n_j}$, and consider the function*

$$\mathrm{BL}(B,p;A) := \left(\frac{\prod_{j\in[m]} \det(A_j)^{p_j}}{\det(\sum_{j\in[m]} p_j B_j^\top A_j B_j)} \right)^{1/2}. \tag{2}$$

Then, the Brascamp-Lieb constant for (B,p), $\mathrm{BL}(B,p)$ is equal to $\sup_{A\in\times_{j\in[m]}\mathbb{S}_+^{n_j}} \mathrm{BL}(B,p;A)$.

Bennett *et al.* [5] proved the following necessary and sufficient conditions for the feasibility of a Brascamp-Lieb datum.

▶ **Theorem 3** (Feasibility of Brascamp-Lieb Datum [5], Theorem 1.15). *Let (B,p) be a Brascamp-Lieb datum with $B_j \in \mathbb{R}^{n_j\times n}$ for each $j \in [m]$. Then, (B,p) is feasible if and only if following conditions hold:*
1. $n = \sum_{j\in[m]} p_j n_j$, *and*
2. $\dim(V) \leq \sum_{j\in[m]} p_j \dim(B_j V)$ *for any subspace V of $\mathbb{R}^n$.*

Theorem 3 introduces infinitely many linear constraints on p as V varies over different subspaces of $\mathbb{R}^n$. However, there are only finitely many different linear restrictions as $\dim(B_jV)$ can only take integer values from $[n_j]$. Consequently, this theorem implies that P_B is a convex set and, in particular, a polytope. It is referred to as the Brascamp-Lieb polytope, see e.g. [4] the "rank one" case ($n_j = 1$ for all j) and [5] for the general case. Some of the above inequality constraints are tight for any $p \in P_B$ such as the inequality constraints induced by $\mathbb{R}^n$ and the trivial subspace, while others can be strict for some $p \in P_B$. If p lies on the boundary of P_B, then there should be some non-trivial subspaces V such that the induced inequality constraints are tight for p. This leads to the definition of critical subspaces and simple Brascamp-Lieb datums.

▶ **Definition 4** (Critical Subspaces and Simple Brascamp-Lieb Data,[9], [5], Definition 1.12)**.** Let (B, p) be a feasible Brascamp-Lieb datum with $B_j \in \mathbb{R}^{n_j \times n}$ for each $j \in [m]$. Then, a subspace V of $\mathbb{R}^n$ is called *critical* if

$$\dim(V) = \sum_{j \in [m]} p_j \dim(B_j V).$$

(B, p) is called *simple* if there is no non-trivial proper subspace of $\mathbb{R}^n$ which is critical.

For a fixed B, simple Brascamp-Lieb data correspond to points p that lie in the relative interior of the Brascamp-Lieb polytope P_B. One important property of simple Brascamp-Lieb data is that there exists a maximizer for $\mathrm{BL}(B, p; A)$. This was proved by Bennett *et al.* [5] by analyzing Lieb's formulation (2). This analysis also leads to a characterization of maximizers of $\mathrm{BL}(B, p; A)$.

▶ **Theorem 5** (Characterization of Maximizers[5], Theorem 7.13)**.** *Let (B, p) be a Brascamp-Lieb datum with $B_j \in \mathbb{R}^{n_j \times n}$ and $p_j > 0$ for all $j \in [m]$. Let $A := (A_j)_{j \in [m]}$ be an m-tuple of positive semidefinite matrices with $A_j \in \mathbb{R}^{n_j \times n_j}$ and let $M := \sum_{j \in [m]} p_j B_j^\top A_j B_j$. Then, the following statements are equivalent,*

1. *A is a global maximizer for $\mathrm{BL}(B, p; A)$ as in (2).*
2. *A is a local maximizer for $\mathrm{BL}(B, p; A)$.*
3. *M is invertible and $A_j^{-1} = B_j M^{-1} B_j^\top$ for each $j \in [m]$.*

Furthermore, the global maximizer A for $\mathrm{BL}(B, p; A)$ exists and is unique up to scalar if and only if (B, p) is simple.

Computational Aspects of the Brascamp-Lieb Inequality. One of the computational questions concerning the Brascamp-Lieb inequality is: Given a Brascamp-Lieb datum (B, p), can we compute $\mathrm{BL}(B, p)$ in time that is polynomial in the number of bits required to represent the datum? Since computing $\mathrm{BL}(B, p)$ exactly may not be possible due to the fact that this number may not be rational even if the datum (B, p) is, one seeks an arbitrarily good approximation. Formally, given the entries of B and p in binary, and an $\varepsilon > 0$, compute a number Z such that

$$\mathrm{BL}(B, p) \leq Z \leq (1 + \varepsilon)\, \mathrm{BL}(B, p)$$

in time that is polynomial in the combined bit lengths of B and p and $\log 1/\varepsilon$.

There are a few obstacles to this problem: (a) Checking if a given Brascamp-Lieb datum is feasible is not known to be in **P**. (b) The formulation of the Brascamp-Lieb constant by Lieb [15] as in (2) is neither concave nor log-concave. Thus, techniques developed in the context of linear and convex optimization do not seem to be directly applicable.

A step towards the computability of $\mathrm{BL}(B, p)$ was taken recently by Garg *et al.* [12] where they presented a pseudo-polynomial time algorithm for (a) and a pseudo-polynomial time algorithm to compute $\mathrm{BL}(B, p)$. The running time of this algorithm to compute $\mathrm{BL}(B, p)$ up to multiplicative error $1 + \varepsilon$ has a polynomial dependency to ε^{-1} and the *magnitude* of the denominators in the components of p rather than the number of bits required to represent them. Garg *et al.* presented a reduction of the problem of computing $\mathrm{BL}(B, p)$ to the problem of computing the "capacity" in an "operator scaling" problem considered by Gurvits [13]. Roughly, in the operator scaling problem, given a representation of a linear mapping from PSD matrices to PSD matrices, the goal is to compute the minimum "distortion" of this mapping; see Section A for their reduction from Brascamp-Lieb to operator scaling. The

operator scaling problem is also not a concave or log-concave optimization problem. However, operator scaling is known to be "geodesically" log-concave; see [1].

Geodesic convexity is an extension of convexity with respect to straight lines in Euclidean spaces to geodesics in Riemannian manifolds. Since all the problems mentioned so far are defined on positive definite matrices, the natural manifold to consider is the space of positive definite matrices with a particular Riemannian metric: the Hessian of the $-\log\det$ function; see section 2. Geodesics are analogs of straight lines on a manifold and, roughly, a function f on is said to be geodesically convex if the average of its values at the two endpoints of any geodesic is at least its value at its mid-point.

The reduction of Garg *et al.* [12], thus, leads to a geodesically log-concave formulation to compute $\mathrm{BL}(B,p)$. However, this construction does not lead to an optimization problem whose dimension is polynomial in the input bit length as the size of constructed positive linear operator in the operator scaling problem depends exponentially on the bit lengths of the entries of p. More precisely, if $p_j = {}^{c_j}/_{c}$ for integers $(c_j)_{j\in[m]}$ and c, then the aforementioned construction results in operators over the space of real-valued, symmetric, positive definite (PD) matrices of dimension $(nc)\times(nc)$, $\mathbb{S}^{nc}_{++}$.

Our Contribution. Our first result is that Lieb's formulation presented in Theorem 2 is jointly geodesically log-concave with respect to inputs $(A_j)_{j\in[m]}$.

▶ **Theorem 6** (Geodesic Log-Concavity of Lieb's Formulation)**.** *Let (B,p) be a feasible Brascamp-Lieb datum with $B_j \in \mathbb{R}^{n_j\times n}$ for each $j\in[m]$. Then,* $\mathrm{BL}(B,p;A)$ *is jointly geodesically log-concave with respect to $A := (A_j)_{j\in[m]}$ where* $\mathrm{BL}(B,\,p;\,A)$ *is defined in* (2)*.*

This formulation leads to a geodesically convex optimization problem on $\bigtimes_{j\in[m]}\mathbb{S}^{n_j}_{++}$ that captures the Brascamp-Lieb constant.

Subsequently, we present a new formulation for the Brascamp-Lieb constant by combining Lieb's result with observations made by Bennett *et al.* [5] about maximizers of $\mathrm{BL}(B,\,p;\,A)$; see Theorem 5. [5] showed that if $A=(A_j)_{j\in[m]}$ is a maximizer to (2), then $A_j = (B_jM^{-1}B_j^\top)^{-1}$ for each $j\in[m]$, where

$$M := \sum_{j\in[m]} p_jB_j^\top A_jB_j.$$

Thus, we can write each A_j as a function of M and obtain, $2\log(\mathrm{BL}(B,p;A(M)))$ equals

$$\sum_{j\in[m]} p_j\log\det((B_jM^{-1}B_j^\top)^{-1}) - \log\det\left(\sum_{j\in[m]} p_jB_j^\top(B_jM^{-1}B_j^\top)^{-1}B_j\right). \tag{3}$$

One can show that the expressions $\log\det((B_jM^{-1}B_j^\top)^{-1})$ for each $j\in[m]$ and $\log\det\left(\sum_{j\in[m]} p_jB_j^\top(B_jM^{-1}B_j^\top)^{-1}B_j\right)$ are geodesically concave functions of M in the positive definite cone. However, the expression in (3) being a difference, is not geodesically concave with respect to M in general. However, if A is a global maximizer of $\mathrm{BL}(B,p;A)$, then we also have that

$$M = \sum_{j\in[m]} p_jB_j^\top(B_jM^{-1}B_j^\top)^{-1}B_j.$$

Combining these two observations, we obtain the following geodesically concave optimization problem for computing the Brascamp-Lieb constant.

▶ **Definition 7** (A Geodesically Log-Concave Formulation for Computation of the Brascsamp-Lieb Constant)**.** Let (B, p) be a feasible Brascamp-Lieb datum with $B_j \in \mathbb{R}^{n_j \times n}$ for each $j \in [m]$. Let $F_{B,\,p}(X) : \mathbb{S}^n_{++} \to \mathbb{R}$ be defined as follows,

$$F_{B,p}(X) := \log\det(X) - \sum_{j\in[m]} p_j \log\det(B_j X B_j^\top). \tag{4}$$

The following theorem establishes the geodesic concavity of $F_{B,p}$ and its equivalence to $\mathrm{BL}(B, p; A)$.

▶ **Theorem 8** (Properties of $F_{B,p}$)**.** *Let (B, p) be a feasible Brascamp-Lieb datum with $B_j \in \mathbb{R}^{n_j \times n}$ for each $j \in [m]$. The function $F_{B,p}(X)$ as defined in (4) has following properties;*

1. *$F_{B,p}$ is geodesically concave.*
2. *If (B, p) is simple, then $\sup_{X\in\mathbb{S}^n_{++}} F_{B,p}(X)$ is attained. Moreover, if $X^\star$ is a maximizer of $F_{B,p}$, then $\exp(\frac{1}{2}F_{B,p}(X^\star)) = \mathrm{BL}(B, p)$ and $A^\star = ((B_j X^\star B_j^\top)^{-1})_{j\in[m]}$ maximizes $\mathrm{BL}(B, p;\, A^\star)$.*

Our results lead to a natural question: is there a polynomial time algorithm based on techniques from geodesic optimization to compute the Brascamp-Lieb constant. For the case when $n_j = 1$ for all j, or the rank-one case, a polynomial time algorithm to compute the Brascamp-Lieb constant is known; see [23, 25]. This algorithm relies on the observation that the dual of the problem to compute the Brascamp-Lieb constant is the problem of optimizing an entropy maximizing probability distribution on the vertices of the Brascamp-Lieb polytope where the marginals of the probability distribution should correspond to the given point p. This algorithm computes $\mathrm{BL}(B, p)$ up to multiplicative error $1 + \varepsilon$ in $\mathrm{poly}(m, \langle B, p\rangle, \log \varepsilon^{-1})$ where $\langle B, p\rangle$ denotes the bit complexity of B, p. The main technical result is to show that for any p in the polytope, an ε-approximate solution to a function like $F_{B,p}$ can be found in a ball of radius that is polynomial in the bit-complexity of B and p. To extend this rank-one result to a higher rank already seems non-trivial due to a couple of reasons. (a) Lack of a separation oracle to the Brascamp-Lieb polytope in general, and (b) lack of an interpretation of $F_{B,p}$ as an optimization problem over the Brascamp-Lieb polytope. The hope is that our formulation might lead to such an optimization interpretation of the Brascamp-Lieb constant over the Brascamp-Lieb polyhedron and, consequently, lead to polynomial time algorithms following the general approach of [25].

2 The Positive Definite Cone, its Riemannian Geometry, and Geodesic Convexity

The Metric. Consider the set of positive definite matrices $\mathbb{S}^d_{++}$ as a subset of $\mathbb{R}^{d\times d}$ with the inner product $\langle X, Y\rangle := \mathrm{Tr}(X^\top Y)$ for $X, Y \in \mathbb{R}^{d\times d}$. At any point $X \in \mathbb{S}^d_{++}$, the tangent space consists of all $d \times d$ real symmetric matrices. There is a natural metric g on this set that gives it a Riemannian structure: For $X \in \mathbb{S}^d_{++}$ and two symmetric matrices ν, ξ

$$g_X(\nu, \xi) := \mathrm{Tr}(X^{-1}\nu X^{-1}\xi). \tag{5}$$

It is an exercise in differentiation to check that this metric arises as the Hessian of the following function $\varphi : \mathbb{S}^d_{++} \to \mathbb{R}$:

$$\varphi(X) := -\log\det X.$$

Hence, $\mathbb{S}^d_{++}$ endowed with the metric g is not only a Riemannian manifold but a Hessian manifold [21]. The study of this metric on $\mathbb{S}^d_{++}$ goes back at least to Siegel [22]; see also the book of Bhatia [6].

Geodesics on $\mathbb{S}^d_{++}$. If $X, Y \in \mathbb{S}^d_{++}$ and $\gamma : [0,1] \to \mathbb{S}^d_{++}$ is a smooth curve between X and Y, then the arc-length of γ is given by the (action) integral

$$L(\gamma) := \int_0^1 \sqrt{g_{\gamma(t)}\left(\frac{d\gamma(t)}{dt}, \frac{d\gamma(t)}{dt}\right)}dt. \tag{6}$$

The geodesic between X and Y is the unique smooth curve between X and Y with the smallest arc-length [26]. The following theorem asserts that between any two points in $\mathbb{S}^d_{++}$, there is a geodesic that connects them. In other words, $\mathbb{S}^d_{++}$ is a geodesically convex set. Moreover, there exists a closed form expression for the geodesic between two points, a formula that is useful for calculations.

▶ **Theorem 9** (Geodesics on $\mathbb{S}^d_{++}$ [6], Theorem 6.1.6)**.** *For $X, Y \in \mathbb{S}^d_{++}$, the exists a unique geodesic between X and Y, and this geodesic is parametrized by the following equation:*

$$X \#_t Y := X^{1/2}(X^{-1/2} Y X^{-1/2})^t X^{1/2} \tag{7}$$

for $t \in [0,1]$.

Geodesic Convexity. One definition of convexity of a function f in a Euclidean space is that the average of the function at the endpoints of each line in the domain is at least the value of the function at the average point on the line. Geodesic convexity is a natural extension of this notion of convexity from Euclidean spaces to Riemannian manifolds that are geodesically convex. A set in the manifold is said to be geodesically convex if, for every pair of points in the set, the geodesic combining these points lies entirely in the set.

▶ **Definition 10** (Geodesically Convex Sets)**.** A set $S \subseteq \mathbb{S}^d_{++}$ is called geodesically convex if for any $X, Y \in S$ and $t \in [0,1]$, $X \#_t Y \in S$.

A function defined on a geodesically convex set is said to be geodesically convex if the average of the function at the endpoints of any geodesic in the domain is at least the value of the function at the average point on the geodesic.

▶ **Definition 11** (Geodesically Convex Functions)**.** Let $S \subseteq \mathbb{S}^d_{++}$ be a geodesically convex set. A function $f : S \to \mathbb{R}$ is called geodesically convex if for any $X, Y \in \mathbb{S}^d_{++}$ and $t \in [0,1]$,

$$f(X \#_t Y) \leq (1-t) f(X) + t f(Y). \tag{8}$$

f is called geodesically concave if $-f$ is geodesically convex.

An important point regarding geodesic convexity is that a non-convex function might be geodesically convex or vice-versa. In general, one cannot convert a geodesically convex function to a convex function by a change of variables. A well-known example for this is the $\log\det(X)$ function whose concavity a classical result from the matrix calculus. On the other hand, a folklore result is that $\log\det(X)$ is both geodesically convex and geodesically concave on the space of positive definite matrices with the metric (5).

▶ **Proposition 12** (Geodesic Linearity of $\log\det$)**.** *The $\log\det(X)$ function is geodesically linear, i.e, it is both geodesically convex and geodesically concave over $\mathbb{S}^n_{++}$.*

Proof. Let $X, Y \in \mathbb{S}^n_{++}$ and $t \in [0,1]$. Then,

$$\log\det(X\#_t Y) \overset{\text{Theorem 9}}{=} \log\det(X^{1/2}(X^{-1/2}YX^{-1/2})^t X^{1/2})$$
$$= (1-t)\log\det(X) + t\log\det(Y).$$

Therefore, $\log\det(X)$ is a geodesically linear function over the positive definite cone with respect to the metric in (5). ◀

Henceforth, when we mention geodesic convexity, it is with respect to the metric in (5). Geodesically convex functions share some properties with usual convex functions. One such property is the relation between local and global minimizers.

▶ **Theorem 13** (Minimizers of Geodesically Convex Functions [20], Theorem 6.1.1)**.** *Let $S \subseteq \mathbb{S}^d_{++}$ be a geodesically convex set and $f : S \to \mathbb{R}$ be a geodesically convex function. Then, any local minimum point of f is also a global minimum of f. More precisely, if $x^\star := \arg\min_{x\in O} f(x)$ for some open geodesically convex subset O of S, then $f(x^\star) = \inf_{x\in S} f(x)$.*

Geometric Mean of Matrices and Linear Maps. While the function $\log\det(P)$ is geodesically linear, our proof of Theorem 6 relies on the geodesic convexity of $\log\det(\sum_{j\in[m]} p_j B_j^\top A_j B_j)$. A simple but important observation is that, if (B,p) is feasible, then $p_j B_j^\top A_j B_j$ is a strictly positive linear map for each j as proved below.

▶ **Lemma 14** (Strictly Linear Maps Induced by Feasible Brascamp-Lieb Datums)**.** *Let $(B,\, p)$ be a Brascamp-Lieb datum with $B_j \in \mathbb{R}^{n_j\times n}$ for each $j \in [m]$ such that $\sum_{j\in[m]} p_j n_j = n$ and $\dim(\mathbb{R}^n) \le \sum_{j\in[m]} p_j \dim(B_j\mathbb{R}^n)$. Then, $\Phi_j(X) := B_j^\top X B_j$ is a strictly positive linear map for each $j \in [m]$.*

Theorem 3 shows that any feasible Brascamp-Lieb datum satisfies both conditions. Furthermore non-feasible Brascamp-Lieb data can also satisfy these conditions as second condition on Theorem 3 enforced for only the $\mathbb{R}^n$.

Proof. Let us assume that for some $j_0 \in [m]$, $\Phi_{j_0}(X)$ is not a strictly positive linear map. Then, there exists $X_0 \in \mathbb{S}^n_{++}$ such that $\Phi_{j_0}(X_0)$ is not positive definite. Thus, there exists $v \in \mathbb{R}^{n_{j_0}}$ such that $v^\top \Phi_{j_0}(X_0) v \le 0$. Equivalently, $(B_{j_0}^\top v)^\top X_0 (B_{j_0}^\top v) \le 0$. Since X_0 is positive definite, we get $B_{j_0}^\top v = 0$. Hence, $v^\top B_{j_0} B_{j_0}^\top v = 0$. Consequently, the rank of B_{j_0} is at most $n_{j_0} - 1$ and $\dim(B_{j_0}\mathbb{R}^n) < n_{j_0}$. Therefore,

$$n = \dim(\mathbb{R}^n) \le \sum_{j\in[m]} p_j \dim(B_j\mathbb{R}^n) < \sum_{j\in[m]} p_j n_j = n,$$

by the hypothesis, a contradiction. Consequently, for any $j \in [m]$, $\Phi_j(X) := B_j^\top X B_j$ is strictly positive linear whenever (B,p) satisfies conditions $\sum_{j\in[m]} p_j n_j = n$ and $\dim(\mathbb{R}^n) \le \sum_{j\in[m]} p_j \dim(B_j\mathbb{R}^n)$. ◀

The joint geodesic convexity of $\log\det(\sum_{j\in[m]} p_j B_j^\top A_j B_j)$ follows from a more general observation (that we prove) that asserts that if Φ_js are strictly positive linear maps from $\mathbb{S}^{n_j}_+$ to $\mathbb{S}^n_+$, then $\log\det(\sum_{j\in[m]} \Phi_j(A_j))$ is geodesically convex. Sra and Hosseini [24] observed this when $m = 1$. Their result also follows from a result of Ando [2] about "geometric means" that is also important for us and we explain it next.

The geometric mean of two matrices was introduced by Pusz and Woronowicz [19]. If $P, Q \in \mathbb{S}^d_{++}$, then the geometric mean of P and Q is defined as

$$P\#_{1/2}Q = P^{1/2}(P^{-1/2}QP^{-1/2})^{1/2}P^{1/2}. \tag{9}$$

By abuse of notation, we drop $1/2$ and denote geometric mean by $P\#Q$. Recall that, the geodesic convexity of a function $f : \mathbb{S}^d_{++} \to \mathbb{R}$ is equivalent to for any $P, Q \in \mathbb{S}^d_{++}$ and $t \in [0, 1]$,

$$f(P\#_t Q) \leq (1-t)f(P) + tf(Q).$$

If f is continuous, then the geodesic convexity of f can be deduced from the following:

$$\forall P, Q \in \mathbb{S}^d_{++}, \quad f(P\#Q) \leq 1/2 f(P) + 1/2 f(Q).$$

Ando proved the following result about the effect of a strictly positive linear map on the geometric mean of two matrices.

▶ **Theorem 15** (Effect of a Linear Map over Geometric Mean [2], Theorem 3). *Let $\Phi : \mathbb{S}^d_+ \to \mathbb{S}^{d'}_+$ be a strictly positive linear map. If $P, Q \in \mathbb{S}^d_{++}$, then*

$$\Phi(P\#Q) \preceq \Phi(P)\#\Phi(Q). \tag{10}$$

The monotonicity of logdet ($P \preceq Q$ implies $\log\det(P) \leq \log\det(Q)$) and the multiplicativity of the determinant, combined with Theorem 15, imply the following result.

▶ **Corollary 16** (Geodesic Convexity of the Logarithm of Linear Maps[24], Corollary 12). *If $\Phi : \mathbb{S}^d_+ \to \mathbb{S}^{d'}_+$ is a strictly positive linear map, then $\log\det(\Phi(P))$ is geodesically convex.*

While the proof of Theorem 8 uses Theorem 16, it is not enough for the proof of Theorem 6. Instead of geodesic convexity of $\log\det(\Phi(P))$, the joint geodesic convexity of $\log\det(\sum_{j\in[m]} \Phi_j(P_j))$ is needed where $\Phi_j : \mathbb{S}^{n_j}_+ \to \mathbb{S}^n_+$ is a linear map for each $j \in [m]$. We conclude this section with the following two results on a maximal characterization of the geometric mean and the effect of positive linear maps on positive definiteness of block diagonal matrices.

▶ **Theorem 17** (Maximal Characterization of the Geometric Mean, see e.g. [6], Theorem 4.1.1). *Let $P, Q \in \mathbb{S}^d_{++}$. The geometric mean of P and Q can be characterized as follows,*

$$P\#Q = \max\left\{ Y \in \mathbb{S}^d_{++} \;\middle|\; \begin{bmatrix} P & Y \\ Y & Q \end{bmatrix} \succeq 0 \right\},$$

where the maximal element is with respect to Loewner partial order.

▶ **Proposition 18** (Effect of Positive Linear Maps, see e.g. [6], Exercise 3.2.2). *Let $\Phi : \mathbb{S}^d_+ \to \mathbb{S}^{d'}_+$ be a strictly positive linear map and $P, Q, R \in \mathbb{S}^d_+$. If*

$$\begin{bmatrix} P & R \\ R & Q \end{bmatrix} \succeq 0, \text{ then } \begin{bmatrix} \Phi(P) & \Phi(R) \\ \Phi(R) & \Phi(Q) \end{bmatrix} \succeq 0.$$

3 Proof of Theorem 6

Let (B, p) be a feasible Brascamp-Lieb datum with $B_j \in \mathbb{R}^{n_j \times n}$ and $A := (A_j)_{j\in[m]}$ with $A_j \in \mathbb{S}^{n_j}_{++}$ be the input of $\mathrm{BL}(B, p; A)$ as defined in (2). To prove the joint geodesic convexity of $\mathrm{BL}(B, p; A)$ with respect to A we extend Theorem 16 and Theorem 15 from linear maps to "jointly linear maps". We use the term jointly linear maps to refer to multivariable functions of the form $\sum_{j\in[m]} \Phi_j(P_j)$ where each Φ_j is a strictly positive linear map for each $j \in [m]$. In particular, the term $\sum_{j\in[m]} p_j B_j^\top A_j B_j$ in (2) is a jointly linear map.

The extension of Theorem 15 is presented in Theorem 20 and its proof is based on the maximal characterization of geometric mean (Theorem 17) and the effect of positive linear maps on the positive definiteness of block matrices (Theorem 18). We follow the proof of Theorem 15 for each Φ_j, but instead of concluding $\Phi_j(P_j \# Q_j) \preceq \Phi_j(P_j) \# \Phi_j(Q_j)$ from the maximality of geometric mean, we sum the resulting inequalities. Subsequently, Theorem 20 follows from the maximality of geometric mean. Lemma 21 is an extension of Theorem 16 and follows directly from Theorem 20.

▶ **Definition 19** (Jointly linear map)**.** Let $\Phi : \mathbb{S}_+^{n_1} \times \cdots \times \mathbb{S}_+^{n_m} \to \mathbb{S}_+^n$. We say that Φ is a jointly linear map if there exist strictly positive linear maps $\Phi_j : \mathbb{S}_+^{n_j} \to \mathbb{S}_+^n$ such that

$$\Phi(P_1, \ldots, P_k) := \sum_{j \in [k]} \Phi_j(P_j). \tag{11}$$

Now, we state the extension of Theorem 15.

▶ **Theorem 20** (Effect of Jointly Linear Maps over Geometric Means)**.** *Let $\Phi : \mathbb{S}_+^{n_1} \times \cdots \times \mathbb{S}_+^{n_m} \to \mathbb{S}_+^n$ be a jointly linear map. Then,*

$$\Phi(G) \preceq \Phi(P) \# \Phi(Q)$$

where $P := (P_j)_{j \in [m]}$, $Q := (Q_j)_{j \in [m]}$, and $G := (G_j)_{j \in [m]}$ with $P_j, Q_j \in \mathbb{S}_{++}^{n_j}$ and $G_j := P_j \# Q_j$.

The following is a corollary of the theorem above and a generalization of Theorem 16.

▶ **Corollary 21** (Joint Geodesic Convexity of Logarithm of Jointly Linear Maps)**.** *If $\Phi : \mathbb{S}_+^{n_1} \times \cdots \times \mathbb{S}_+^{n_m} \to \mathbb{S}_+^n$ is a jointly linear map, then*

$$g(P_1, \ldots, P_m) := \log\det(\Phi(P_1, \ldots, P_m)) \tag{12}$$

is jointly geodesically convex in $\mathbb{S}_{++}^{n_1} \times \cdots \times \mathbb{S}_{++}^{n_m}$.

Proof of Corollary 21. We show that g is jointly geodesically mid-point convex. Theorem 20 implies that

$$\Phi(G) \preceq \Phi(P) \# \Phi(Q) \tag{13}$$

for any $P := (P_j)_{j \in [m]}$ and $Q := (Q_j)_{j \in [m]}$ with $P_j, Q_j \in \mathbb{S}_{++}^{n_j}$. Therefore,

$$\begin{aligned} g(G) &\overset{(12)}{=} \log\det(\Phi(G)) \\ &\leq \log\det(\Phi(P) \# \Phi(Q)) && \text{(monotonicity of } \log\det \text{ and (13))} \\ &\leq {}^1\!/_2 \log\det(\Phi(P)) + {}^1\!/_2 \log\det(\Phi(Q)) && \text{(multiplicativity of det)} \\ &\overset{(12)}{=} {}^1\!/_2 \left(g(P) + g(Q)\right). \end{aligned}$$

Thus, g satisfies mid-point geodesic convexity. Consequently, we establish the geodesic convexity of g using the continuity of g. ◀

The proof of Theorem 6 is a simple application of Corollary 21 and Theorem 12.

Proof of Theorem 6. We show that $\mathrm{BL}(B, p; A)$ is jointly geodesically mid-point log-concave with respect to A. In other words, we show that for arbitrary $P = (P_j)_{j \in [m]}$, $Q = (Q_j)_{j \in [m]}$

$$-\log \mathrm{BL}(B, p; G) \leq -{}^1\!/_2 \left(\log \mathrm{BL}(B, p; P) + \log \mathrm{BL}(B, p; Q)\right)$$

where $G = (G_j)_{j\in[m]}$ with $G_j := P_j \# Q_j$, being the midpoint of geodesic combining P_j to Q_j. This implies that $\mathrm{BL}(B,p;A)$ is jointly geodesically log-concave with respect to A due to the continuity of $\mathrm{BL}(B,p;A)$ with respect to A.

Let $\Phi_j(P_j) := p_j B_j^\top P_j B_j$. Φ_j is strictly positive linear map by Lemma 14. Then, $\Phi(P) := \sum_{j\in[m]} p_j B_j^\top P_j B_j$ is jointly linear, as $\Phi(P) = \sum_{j\in[m]} \Phi_j(P_j)$. Hence, $\log\det(\Phi(P))$ is jointly geodesically convex by Corollary 21. Also, $\log\det(X)$ is geodesically linear (Theorem 12). Thus, for any $P := (P_j)_{j\in[m]}$, $Q := (Q_j)_{j\in[m]}$ and $G := (G_j)_{j\in[m]}$ with $G_j := P_j \# Q_j$ we have

$$\begin{aligned}
-\log \mathrm{BL}(B,p;G) &\overset{(2)}{=} 1/2(\log\det(\Phi(G)) - \sum_{j\in[m]} p_j \log\det(G_j)) \\
&\overset{(12)}{\leq} 1/2(1/2(\log\det(\Phi(P)) + \log\det(\Phi(Q))) - \sum_{j\in[m]} p_j \log\det(G_j)) \\
&= 1/2(1/2(\log\det(\Phi(P)) - \sum_{j\in[m]} p_j \log\det(P_j)) \qquad \text{(Theorem 12)} \\
&\qquad + 1/2(\log\det(\Phi(Q)) - \sum_{j\in[m]} p_j \log\det(Q_j))) \\
&\overset{(2)}{=} -1/2(\log\det \mathrm{BL}(B,p;P) + \log\det \mathrm{BL}(B,p;Q)).
\end{aligned}$$

This concludes the proof. ◀

Now we prove Theorem 20. This proof is based on the proof of Theorem 15 and depends on the maximality of geometric mean (Theorem 17) and effects of positive linear maps on block matrices (Theorem 18).

Proof of Theorem 20. Φ is a jointly linear map by the assumption. Thus, there exist linear maps $\Phi_j : \mathbb{S}_+^{n_j} \to \mathbb{S}_+^n$ such that $\Phi(P) = \sum_{j\in[m]} \Phi_j(P_j)$. Theorem 17 implies for each $j \in [m]$,

$$0 \preceq \begin{bmatrix} P_j & G_j \\ G_j & Q_j \end{bmatrix}.$$

Since Φ_j's are strictly positive linear maps, Theorem 18 implies that for each $j \in [m]$,

$$0 \preceq \begin{bmatrix} \Phi_j(P_j) & \Phi_j(G_j) \\ \Phi_j(G_j) & \Phi_j(Q_j) \end{bmatrix}.$$

The dimension of these block matrices is $2n \times 2n$ for each $j \in [m]$. Thus we can sum these inequalities and the summation leads to

$$0 \preceq \sum_{j\in[m]} \begin{bmatrix} \Phi_j(P_j) & \Phi_j(G_j) \\ \Phi_j(G_j) & \Phi_j(Q_j) \end{bmatrix} = \begin{bmatrix} \sum_{j\in[m]} \Phi_j(P_j) & \sum_{j\in[m]} \Phi_j(G_j) \\ \sum_{j\in[m]} \Phi_j(G_j) & \sum_{j\in[m]} \Phi_j(Q_j) \end{bmatrix} \overset{(11)}{=} \begin{bmatrix} \Phi(P) & \Phi(G) \\ \Phi(G) & \Phi(Q) \end{bmatrix}. \tag{14}$$

Theorem 17 and (14) imply that $\Phi(G) \preceq \Phi(P) \# \Phi(Q)$. ◀

4 Proof of Theorem 8

Let (B,p) be a feasible Brascamp-Lieb datum with $B_j \in \mathbb{R}^{n_j \times n}$. Let $A := (A_j)_{j\in[m]}$ with $A_j \in \mathbb{S}_{++}^{n_j}$ be the input of $\mathrm{BL}(B,p;A)$ as defined in (2). The proof of Theorem 8 first establishes the geodesic concavity of $F_{B,p}$ as defined in (4) when (B,p) is feasible. Next,

it establishes the relation between global maximizers of $F_{B,p}(X)$ and global maximizers of $\mathrm{BL}(B,p;A)$, as well as the relation between $\sup_{X\in\mathbb{S}^n_{++}} F_{B,p}(X)$ and $\mathrm{BL}(B,p)$ when (B,p) is simple.

Feasibility of (B,p) implies the linear maps $B_jXB_j^\top$ are strictly positive linear for each $j\in[m]$ (Lemma 14). Consequently, $-\log\det(B_jXB_j^\top)$ is geodesically concave by Theorem 16 for each $j\in[m]$. Also, $\log\det(X)$ is geodesically concave by Theorem 12. Thus, $F_{B,p}(X)$ is geodesically concave as a sum of geodesically concave functions with non-negative coefficients.

The geodesic concavity of $F_{B,p}$ implies that any local maximum is also a global maximum (Theorem 13). Consequently, we investigate the points where all directional derivatives of $F_{B,p}$ vanish, the critical points of $F_{B,p}$. A simple calculation involving the first derivative shows that any critical point X of $F_{B,p}$ should satisfy

$$X^{-1}=\sum_{j\in[m]} p_jB_j^\top(B_jXB_j^\top)^{-1}B_j.$$

Theorem 5 implies that we can construct a global maximizer of $\mathrm{BL}(B,p;A)$ from X by setting $A_j:=(B_jXB_j^\top)^{-1}$. Furthermore, we can construct a critical point of $F_{B,p}$ using a global maximizer of $\mathrm{BL}(B,p;A)$ by setting $X:=(\sum_{j\in[m]}p_jB_j^\top A_jB_j)^{-1}$. Theorem 5 guarantees the existence of a global maximizer of $\mathrm{BL}(B,p;A)$ if (B,p) is simple. Thus, if (B,p) is simple, then $\sup_X F_{B,p}(X)$ should be attained. We can deduce

$$\sup_X F_{B,p}(X)=2\log\mathrm{BL}(B,p)$$

from the construction of $F_{B,p}$ and the relation between maximizers of $F_{B,p}$ and $\mathrm{BL}(B,p;A)$.

The second part of the proof Theorem 8 depends on well-known identities from matrix calculus. We present these identities for the convenience of the reader and refer the interested readers to the matrix cookbook [18] for more details.

▶ **Proposition 22.** *Let $X(t),Y(t)$ be differentiable functions from $\mathbb{R}$ to $d\times d$ invertible symmetric matrices. Let $U\in\mathbb{R}^{d'\times d}$, $V\in R^{d\times d''}$, $W\in\mathbb{R}^{d\times d}$ be matrices which do not depend on t. Then, the following identities hold:*

$$\frac{d\log\det(X(t))}{dt}=\mathrm{Tr}\left(X(t)^{-1}\frac{dX(t)}{dt}\right) \tag{15}$$

$$\frac{dUX(t)V}{dt}=U\frac{dX(t)}{dt}V \tag{16}$$

$$\frac{dtW}{dt}=W. \tag{17}$$

Proof of Theorem 8. We start by showing that $F_{B,p}(X)$ is geodesically concave. The feasibility of (B,p) implies $B_jXB_j^\top$ is a strictly positive linear map for each $j\in[m]$ (Lemma 14). Thus, Theorem 16 yields that for any $X,Y\in\mathbb{S}^n_{++}$,

$$\log\det(B_j(X\#Y)B_j^\top)\le{}^1\!/_2\log\det(B_jXB_j^\top)+{}^1\!/_2\log\det(B_jYB_j^\top). \tag{18}$$

Combining this with the geodesic linearity of $\log\det(X)$ (Theorem 12), we obtain

$$\begin{aligned}F_{B,p}(X\#Y)=&\log\det(X\#Y)-\sum_{j\in[m]}p_j\log\det(B_j(X\#Y)B_j^\top)\\ \ge&{}^1\!/_2(\log\det(X)-\sum_{j\in[m]}p_j\log\det(B_jXB_j^\top))\\ &+{}^1\!/_2(\log\det(Y)-\sum_{j\in[m]}p_j\log\det(B_jYB_j^\top))\\ =&{}^1\!/_2F_{B,p}(X)+{}^1\!/_2F_{B,p}(Y)\end{aligned}$$

Therefore, $F_{B,p}(X)$ is geodesically concave.

Now, we can show the second part of the theorem. The geodesic concavity of $F_{B,p}$ implies any local maximum of $F_{B,p}$ is a global maximum of $F_{B,p}$. A local maximum of $F_{B,p}$ is achieved at X if it is a critical point of $F_{B,p}$. If X is a critical point of $F_{B,p}$, then for any symmetric matrix Q, the directional derivative of $F_{B,p}$ at X in the direction of Q should be 0. In other words, if $\zeta(t) := X + tQ$ and $f(t) := F_{B,p}(\zeta(t))$, then $\frac{df}{dt}\Big|_{t=0}$ should be 0 for any Q. Let us compute $\frac{df}{dt}$,

$$\begin{aligned}
\frac{df}{dt} &\overset{(4)}{=} \frac{d}{dt}\log\det(\zeta(t)) - \sum_{j\in[m]} p_j \frac{d}{dt}\log\det(B_j\zeta(t)B_j^\top) \\
&\overset{(15)}{=} \mathrm{Tr}\left(\zeta(t)^{-1}\frac{d\zeta(t)}{dt}\right) - \sum_{j\in[m]} p_j \mathrm{Tr}\left((B_j\zeta(t)B_j^\top)^{-1}\frac{dB_j\zeta(t)B_j^\top}{dt}\right) \\
&\overset{(16)}{=} \mathrm{Tr}\left(\zeta(t)^{-1}\frac{d\zeta(t)}{dt}\right) - \sum_{j\in[m]} p_j \mathrm{Tr}\left((B_j\zeta(t)B_j^\top)^{-1}B_j\frac{d\zeta(t)}{dt}B_j^\top\right) \\
&\overset{(17)}{=} \mathrm{Tr}(\zeta(t)^{-1}Q) - \sum_{j\in[m]} p_j \mathrm{Tr}((B_j\zeta(t)B_j^\top)^{-1}B_jQB_j^\top).
\end{aligned}$$

Hence, the directional derivative of $F_{B,p}(X)$ in the direction of Q, $\frac{df}{dt}\Big|_{t=0}$ is

$$\mathrm{Tr}(X^{-1}Q) - \sum_{j\in[m]} p_j \mathrm{Tr}((B_jXB_j^\top)^{-1}B_jQB_j^\top). \tag{19}$$

If directional derivates of $F_{B,p}$ vanish at X, then (19) should be 0 for any symmetric matrix Q. Consequently,

$$\mathrm{Tr}(Q[X^{-1} - \sum_{j\in[m]} p_jB_j^\top(B_jXB_j^\top)^{-1}B_j]) = 0.$$

This observation leads to

$$X^{-1} = \sum_{j\in[m]} p_jB_j^\top(B_jXB_j^\top)^{-1}B_j. \tag{20}$$

If (B,p) is simple, then there exists an input $A^\star = (A_j)_{j\in[m]}$ such that $A_j^{-1} = B_jM^{-1}B_j^\top$ where $M = \sum_{j\in[m]} p_jB_j^\top A_jB_j$ by Theorem 5. Consequently, M satisfies

$$M = \sum_{j\in[m]} p_jB_j^\top(B_jM^{-1}B_j^\top)^{-1}B_j. \tag{21}$$

If $X^\star := M^{-1}$, then $X^\star$ satisfies (20) due to (21). Thus, $X^\star$ is a critical point of $F_{B,p}(X)$ and $F_{B,p}$ attains its maximal value at $X^\star$. Furthermore, the maximizer $A^\star$ of $\mathrm{BL}(B,p;A)$ is equal to $((B_jX^\star B_j^\top)^{-1})_{j\in[m]}$. Finally,

$$\begin{aligned}
F_{B,p}(X^\star) &\overset{(4)}{=} \log\det(X^\star) - \sum_{j\in[m]} p_j\log\det(B_jX^\star B_j^\top) \\
&= \log\det((\sum_{j\in[m]} p_jB_j^\top(B_jX^\star B_j^\top)^{-1}B_j)^{-1}) - \sum_{j\in[m]} p_j\log\det(B_jX^\star B_j^\top) \\
&= \log\det((\sum_{j\in[m]} p_jB_j^\top A_jB_j)^{-1}) - \sum_{j\in[m]} p_j\log\det(A_j^{-1}) \\
&= \sum_{j\in[m]} p_j\log\det(A_j) - \log\det(\sum_{j\in[m]} p_jB_j^\top A_jB_j).
\end{aligned}$$

Therefore, $\mathrm{BL}(B,p;A^\star) = \exp(\frac{1}{2}F_{B,p}(X^\star))$. ◀

References

1. Zeyuan Allen-Zhu, Garg Ankit, Yuanzhi Li, Rafael Oliveira, and Avi Wigderson. Operator scaling via geodesically convex optimization, invariant theory and polynomial identity testing. In *Proceedings of the 50th Annual ACM Symposium on Theory of Computing, Los Angeles, CA, USA, June 25-29*, 2018. To appear. arXiv:arXiv:1804.01076.
2. Tsuyoshi Ando. Concavity of certain maps on positive definite matrices and applications to Hadamard products. *Linear Algebra and its Applications*, 26:203–241, 1979. doi:10.1016/0024-3795(79)90179-4.
3. Keith Ball. Volumes of sections of cubes and related problems. In *Geometric Aspects of Functional Analysis*, pages 251–260. Springer, 1989.
4. Franck Barthe. On a reverse form of the Brascamp-Lieb inequality. *Inventiones mathematicae*, 134(2):335–361, 1998.
5. 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.
6. Rajendra Bhatia. *Positive definite matrices*. Princeton University Press, 2009.
7. Herm Jan Brascamp and Elliott H Lieb. Best constants in Young's inequality, its converse, and its generalization to more than three functions. *Advances in Mathematics*, 20(2):151–173, 1976.
8. Eric A Carlen and Dario Cordero-Erausquin. Subadditivity of the entropy and its relation to Brascamp–Lieb type inequalities. *Geometric and Functional Analysis*, 19(2):373–405, 2009.
9. Eric A. Carlen, Elliott. H. Lieb, and Michael Loss. A sharp analog of Young's inequality on SN and related entropy inequalities. *The Journal of Geometric Analysis*, 14(3):487–520, Sep 2004. doi:10.1007/BF02922101.
10. Zeev Dvir, Ankit Garg, Rafael Mendes de Oliveira, and József Solymosi. Rank bounds for design matrices with block entries and geometric applications. *Discrete Analysis*, 2018:5, Mar 2018. doi:10.19086/da.3118.
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, Los Angeles, CA, USA, May 31- June 3, 2014*, pages 784–793, New York, NY, USA, 2014. doi:10.1145/2591796.2591818.
12. Ankit Garg, Leonid Gurvits, Rafael Mendes de Oliveira, and Avi Wigderson. Algorithmic and optimization aspects of Brascamp-Lieb inequalities, via operator scaling. In *Proceedings of the 49th Annual ACM Symposium on Theory of Computing, Montreal, QC, Canada, June 19-23, 2017*, pages 397–409, 2017. doi:10.1145/3055399.3055458.
13. Leonid Gurvits. Classical complexity and quantum entanglement. *Journal of Computer and System Sciences*, 69(3):448–484, 2004. Special Issue on STOC 2003. doi:10.1016/j.jcss.2004.06.003.
14. Moritz Hardt and Ankur Moitra. Algorithms and hardness for robust subspace recovery. In *Conference on Learning Theory*, pages 354–375, 2013.
15. Elliott H Lieb. Gaussian kernels have only Gaussian maximizers. *Inventiones Mathematicae*, 102(1):179–208, 1990.
16. Jingbo Liu, Thomas A. Courtade, Paul W. Cuff, and Sergio Verdú. Smoothing Brascamp-Lieb inequalities and strong converses for common randomness generation. In *IEEE International Symposium on Information Theory, Barcelona, Spain, July 10-15, 2016*, pages 1043–1047. IEEE, 2016. doi:10.1109/ISIT.2016.7541458.
17. Jingbo Liu, Thomas A. Courtade, Paul W. Cuff, and Sergio Verdú. Information-theoretic perspectives on Brascamp-Lieb inequality and its reverse. *CoRR*, abs/1702.06260, 2017. arXiv:1702.06260.

18 Kaare Brandt Petersen, Michael Syskind Pedersen, et al. The matrix cookbook. *Technical University of Denmark*, 7(15):510, 2008.

19 W. Pusz and S.L. Woronowicz. Functional calculus for sesquilinear forms and the purification map. *Reports on Mathematical Physics*, 8(2):159–170, 1975. doi:10.1016/0034-4877(75)90061-0.

20 Tamás Rapcsák. *Geodesic convex functions*, pages 61–86. Springer US, Boston, MA, 1997. doi:10.1007/978-1-4615-6357-0_6.

21 Hirohiko Shima. *The geometry of Hessian structures*. World Scientific Publishing, 2007.

22 Carl Ludwig Siegel. Symplectic geometry. *American Journal of Mathematics*, 65(1):1–86, 1943. URL: http://www.jstor.org/stable/2371774.

23 Mohit Singh and Nisheeth K Vishnoi. Entropy, optimization and counting. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 50–59. ACM, 2014.

24 Suvrit Sra and Reshad Hosseini. Conic geometric optimization on the manifold of positive definite matrices. *SIAM Journal on Optimization*, 25(1):713–739, 2015.

25 Damian Straszak and Nisheeth K. Vishnoi. Computing maximum entropy distributions everywhere. *ArXiv e-prints*, 2017. arXiv:1711.02036.

26 Nisheeth K Vishnoi. Geodesic convex optimization: Differentiation on manifolds, geodesics, and convexity, Jun 2018. URL: https://nisheethvishnoi.files.wordpress.com/2018/06/geodesicconvexity.pdf.

A An Exponential-Sized Geodesically Convex Formulation from Operator Scaling

In this section, we describe the operator scaling problem and the reduction of Garg *et al.* [12] from the computation of the Brascasmp-Lieb constant to the computation of the "capacity" of a positive operator.

The Operator Scaling Problem and its Geodesic Convexity. In the operator scaling problem [13], one is given a linear operator $T(X) := \sum_{j\in[m]} T_j^\top X T_j$ through the tuple of matrices T_js and the goal is to find square matrices L and R such that

$$\sum_{j\in[m]} \hat{T}_j^\top \hat{T}_j = I \qquad \text{and} \qquad \sum_{j\in[m]} \hat{T}_j \hat{T}_j^\top = I, \tag{22}$$

where $\hat{T}_j := L T_j R$. The matrices L and R can be computed by solving the following optimization problem.

▶ **Definition 23** (Operator Capacity)**.** Let $T : \mathbb{S}^d_{++} \to \mathbb{S}^{d'}_{++}$ be a linear operator, then the capacity of T is

$$\operatorname{cap}(T) := \inf_{\det(X)=1} \det\left(\frac{d'}{d} T(X)\right). \tag{23}$$

In particular, if $X_T^\star$ is a minimizer of (23) and $Y_T^\star = T(X_T^\star)^{-1}$, then (22) holds if we let $L := (Y_T^\star)^{1/2}$ and $R := (X_T^\star)^{1/2}$; see [13] for details. Operator capacity is known to be geodesically log-convex, see e.g. [1], Lemma C.1.

The Reduction. Let (B, p) be a Brascamp-Lieb datum with $B_j \in \mathbb{R}^{n_j \times n}$ for each $j \in [m]$. Garg *et al.* [12] proved that if the exponent $p = (p_j)_{j\in[m]}$ is a rational vector, then one can construct an operator scaling problem from (B, p). Let $p_j = {}^{c_j}/_{c}$ where c_js are non-negative integers and c is a positive integer, the common denominator for all the p_js. Their reduction, outlined below, results in an operator $T_{B,p} : \mathbb{S}^{nc}_{++} \to \mathbb{S}^{n}_{++}$ with the property that $\mathrm{cap}(T_{B,p}) = {}^{1}/_{\mathrm{BL}(B,p)^2}$.

The operator $T_{B,p}$ is constructed with c_j copies of the matrix B_j for each $j \in [m]$. In order to easily refer these copies, let us define $m' := \sum_{j\in[m]} c_j$, and the function $\delta : [m'] \to [m]$. $\delta(i)$ is defined as the integer j such that,

$$\sum_{k<j} c_k < i \leq \sum_{k\leq j} c_k.$$

Let Z_{ij} be an $n_{\delta(i)} \times n$ matrix all of whose entries are zero when $\delta(i) \neq j$ and $B_{\delta(i)}$ if $\delta(i) = j$, for $i, j \in [m']$. Define $nc \times n$ matrices T_j for $j \in [m]$ as follows:

$$T_j := \begin{bmatrix} Z_{1j} \\ \vdots \\ Z_{m'j} \end{bmatrix},$$

and define the linear operator $T_{B,p} : \mathbb{S}^{nc}_{++} \to \mathbb{S}^{n}_{++}$ as

$$T_{B,p}(X) := \sum_{j\in[m']} T_j^\top X T_j. \tag{24}$$

▶ **Theorem 24** (Reduction from Brascamp-Lieb to Operator Scaling[12], Lemma 4.4.)**.** *Let (B, p) be a Brascamp-Lieb a datum with $B_j \in \mathbb{R}^{n_j \times n}$ and $p_j = c_j/c$ where $c, c_j \in \mathbb{Z}_+$ for each $j \in [m]$. The capacity of the operator $T_{B,p}$ defined in* (24) *satisfies* $\mathrm{cap}(T_{B,p}) = {}^{1}/_{\mathrm{BL}(B,p)^2}$.

While this reduction gives a geodesically log-concave formulation to compute the Brascamp-Lieb constant, the dimension of the optimization problem is exponentially large in the bit complexity of the Brascamp-Lieb datum. Consequently, a truly polynomial time algorithm for the computation of the Brascamp-Lieb constant does not follow from any black-box optimization method for geodesically convex functions or polynomial time algorithms for operator capacity; e.g. the algorithm presented in [1].

Tensor Rank is Hard to Approximate

Joseph Swernofsky[1]
Kungliga Tekniska Högskolan, Lindstedtsvägen 3, Stockholm SE-100 44, Sweden
josephsw@kth.se

Abstract

We prove that approximating the rank of a 3-tensor to within a factor of $1 + 1/1852 - \delta$, for any $\delta > 0$, is NP-hard over any field. We do this via reduction from bounded occurrence 2-SAT.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness

Keywords and phrases tensor rank, high rank tensor, slice elimination, approximation algorithm, hardness of approximation

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.26

Acknowledgements I thank Johan Håstad for guidance during research and for thoroughly reviewing and giving advice on the writing of this paper. I thank Per Austrin for discussions and for improving the inapproximability constant.

1 Introduction

The rank of a matrix is well understood and easy to compute. The student first introduced to the notion of rank often learns that they can perform Gaussian elimination on a matrix and the number of nonzero rows they obtain is its rank. They may even learn that this is equivalent to the number of linearly independent rows of the matrix. A rank 1 matrix can be written as an outer product of two vectors u and v, meaning its ij^{th} entry is $u_i \cdot v_j$. The rank of M is equivalent to the minimum number of outer products (rank 1 matrices) that must be added to form M.

This notion of rank has been generalized to describe tensors. A 3-tensor T is a grid of numbers with 3 indices. A tensor is rank 1 if it is the outer product of 3 vectors, and the rank of T is the minimum number of rank 1 tensors that must be added to form it.

Rank of 3-tensors is, apart from its inherent mathematical appeal, a natural tool for reasoning about bilinear circuit complexity. The inputs and outputs of a bilinear circuit correspond to indices in the 3 coordinates of a 3-tensor, and the rank is equal to the number of multiplication nodes needed in the circuit [8]. Strassen's [17] famous algorithm for 2×2 matrix multiplication can be viewed in this light. The multiplication is a circuit computing the entries of the output matrix from the entries of the input matrices. Writing this as a $2^2 \times 2^2 \times 2^2$ tensor, his basic construction corresponds to a rank 7 decomposition. He then applied this recursively to create the first algorithm for $n \times n$ matrix multiplication that was faster than the naive algorithm.

Unfortunately, while matrix rank is easy to compute, Håstad [6] showed that tensor rank is NP-hard to calculate. More recently, Shitov [14] and Schaefer and Štefankovič [13] showed that rank over a field $\mathbb{F}$ is complete for the existential theory of $\mathbb{F}$ and is also uncomputable over $\mathbb{Z}$. This presents a roadblock for researchers hoping to analyze the rank of tensors for

[1] Supported by the Knut and Alice Wallenberg Foundation.

specific applications. Much work has been done since Strassen's initial paper to analyze larger tensors modeling matrix multiplication and many improvements have been obtained. However, even very simple tensors stubbornly resist analysis, such as the $3^2 \times 3^2 \times 3^2$ tensor for 3×3 matrix multiplication. Its rank is only known to be between 19 [3] and 23 [9].

Another application for tensor rank is demonstrated by Raz [12], who shows how strong circuit lower bounds for more general circuits may be possible with a better understanding of rank. Tensor rank is also a practical tool in a variety of fields, such as signal processing and computer vision [7]. Our knowledge is very limited though, because despite knowing that there are 3-tensors with rank at least $n^2/3$ (which can be seen by dimension counting), the highest rank known for an explicit family of 3-tensors is $O(n)$ [15]. Also, while we know rank is hard to compute exactly, we do not know if it can be approximated. Alexeev et al. [1] and Bläser [4] both mention this open question. Recently Song et al. [16] proved that, assuming the Exponential Time Hypothesis (ETH), there is some constant c_0 so that tensor rank cannot be approximated within a factor of c_0 in polynomial time. We strengthen this to prove an NP-hardness result and in particular the following theorem.

▶ **Theorem 1.** *It is NP-hard to approximate 3-tensor rank over any field $\mathbb{F}$ within a factor of $1 + 1/1852 - \delta$, for any $\delta > 0$.*

Our construction is a re-analysis and simplification of the reduction by Håstad [6]. We show that if bounded occurrence SAT is used as the starting point for the reduction then significant extra rank can be guaranteed from unsatisfied clauses over the rank in the satisfiable case. Independently, Bläser et al. [5] proved the same result, but without an explicit constant, and with a slightly more involved argument.

A natural follow-up question to ask is whether tensor rank is hard to approximate within any constant, or within a specific unbounded function. It would also be interesting to have any nontrivial approximation algorithm. We discuss these questions briefly in Section 4.

2 Preliminaries

We work over an arbitrary field $\mathbb{F}$ throughout this paper.

A ***d*-tensor** is a function from d-tuples of natural numbers to $\mathbb{F}$. The entries in the tuple represent coordinates in a d-dimensional space. A given d-tensor has size n_i in coordinate i and is defined at exactly those s where $s_i \in [n_i]$ for all i. It is helpful to think of this as a grid with numbers written in the cells. We typically write T_s to denote $T(s)$, or even $T_{s_1 s_2 \ldots s_d}$. For $v_i \in \mathbb{F}^{n_i}$, an outer product $\otimes_{i \in [d]} v_i$ of d vectors is a rank 1 tensor. The **rank, $\mathbf{rk}(\boldsymbol{T})$,** is the minimum r so that $T = \sum_{j \in [r]} \otimes_{i \in [d]} v_i^j$.

A minimization problem P consists of a set of valid strings L together with a score function $s : L \to \mathbb{R}^+$. An **approximation algorithm** for P is an algorithm A that takes strings in L and outputs numbers in $\mathbb{R}^+$ with $\forall x \in L. A(x) \geq s(x)$. Let L_n be the strings in L of length n. For a function $c : \mathbb{N} \to \mathbb{R}^+$, A is a c-approximation algorithm if $\max_{x \in L_n} \frac{A(x)}{s(x)} \leq c(n)$.

A **MAX-E2-SAT** instance consists of a set of variables $\mathbf{x} = \{x_k\}_{k \in [n]}$ and a set of disjunctive clauses $\{l_{i1} \vee l_{i2}\}_{i \in [m]}$ of size exactly 2 over those variables. Here $n, m \in \mathbb{N}$ and l_{ij} is a literal of a variable in $\mathbf{x}$ for each $i \in [m], j \in [2]$. The problem is to compute the maximum number of clauses that can be simultaneously satisfied by an assignment to $\mathbf{x}$. **E3-OCC-MAX-2SAT** is MAX-E2-SAT where every variable occurs in exactly 3 clauses. Note that in E3-OCC-MAX-2SAT we have $m = \frac{3n}{2}$.

2.1 Slices

A **slice** of a tensor is the grid obtained from fixing one of the coordinates. The slice $T_{i:x}$ of a d-tensor T is a $(d-1)$-tensor with $T_{i:x}(s) := T(s_1, s_2, \dots s_{i-1}, x, s_i, \dots s_{d-1})$. That is, when indexing into $T_{i:x}$ one is indexing into T but omitting the coordinate i. For example, if $d = 3$ then $T_{3:k}$ is the matrix M where $M_{ij} = T_{ijk}$. Even more concretely, if we draw the first coordinate as the row and the second coordinate as the column:

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}, A_{2:1} = \begin{bmatrix} a \\ c \end{bmatrix}$$

We want to recover some of our intuition about matrix rank and use row reduction when computing tensor rank.

▶ **Lemma 2.** *If a slice is scaled by a nonzero constant or added to another slice in the same coordinate, the rank does not change.*

Proof. Suppose we have a rank r decomposition $T = \sum_{h\in[r]} \otimes_{j\in[d]} v_j^h$. If slice $T_{i:x}$ is scaled by λ, simply replace $v_i^h(x)$ with $\lambda v_i^h(x)$ for every h. This shows that the rank does not increase. Given a rank r' decomposition of the new tensor, scale by $1/\lambda$ to see that $r' \geq r$.

To add $T_{i:x}$ to $T_{i:y}$, simply replace $v_i^h(y)$ with $v_i^h(y) + v_i^h(x)$. Again, this operation can be inverted by replacing $v_i^h(y)$ with $v_i^h(y) - v_i^h(x)$, so the ranks before and after adding $T_{i:x}$ to $T_{i:y}$ are the same. ◀

▶ **Corollary 3.** *If a slice is a linear combination of other slices in the same coordinate, setting every entry to 0 or removing it does not change the rank.*

We call the process of iteratively removing dependent slices until all remaining slices are independent **slice elimination**. When performing row reduction on a matrix, one can add any row u to any other row v without changing the rank. Gaussian elimination turns this into a simple strategy for calculating rank where a row u is added, scaled appropriately, to each other row according to a simple rule. One can think of this as choosing some vector c to extend u by, forming the outer product $u \otimes c$, and adding this to the rank decomposition of the matrix.

Unfortunately, slice elimination does not give an efficient algorithm for tensor rank for two reasons. First, unlike in a matrix, the non-eliminated slices can have rank greater than 1 and it will be unclear what the overall rank of the tensor is. Second, even choosing what multiples to use when adding one slice to the others is NP-hard, and this is the property exploited in showing NP-hardness of tensor rank.

2.2 Substitution

While slice elimination does not give an algorithm for tensor rank, it can be used to analyze the rank of some tensors. A rank 1 slice T' of T can be written uniquely as an outer product, so it is natural to assume T' appears in some minimum rank decomposition of T. Indeed, we show this can be assumed in the following lemma. A form of this for polynomials comes from Pan [11] and goes by many names, such as “slice reduction", “layer reduction", and “substitution". We slightly generalize the proof by Håstad [6]. More general versions can be found in papers by Alexeev et al. [1] and Landsberg and Michałek [10]:

▶ **Lemma 4** (Substitution)**.** *Given a d-tensor T of rank r, suppose the 1-slices $T_{1:j}$ for $j \in [k]$ are rank 1 and linearly independent as vectors. Then there is an expansion $T = \sum_{j=1}^{r} v^j \otimes M^j$ with $\mathrm{rk}(M^j) = 1$ and $M^j = T_{1:j}$ for $j \in [k]$.*

Note that the lemma is stated using 1-slices and using coordinates $j \in [k]$ for $T_{1:j}$ for simplicity. It also holds when taking slices in an arbitrary coordinate i and considering an arbitrary list $S \subset [n_i]$ of i-coordinates.

Proof. Suppose $k = 1$ and write $T_{1:1}$ as a linear combination of M^j. We pick one of the M^j and rearrange the equation to write it in terms of $T_{1:1}$ and the other $M^{j'}$. Then we substitute this into the equations for the other slices, thereby eliminating the use of M^j and introducing $T_{1:1}$.

Explicitly, the rank decomposition gives us

$$T_{1:x} = \sum_{j \in [r]} v^j(x) M^j$$

Assume without loss of generality that $v^1(1) \neq 0$. We use this equation to replace all occurrences of M^1 with $T_{1:1}$. First we rearrange the equation for $T_{1:1}$ to get

$$M^1 = \frac{1}{v^1(1)} T_{1:1} - \sum_{j=2}^{r} \frac{v^j(1)}{v^1(1)} M^j$$

Now we can substitute this in the remaining equations and simplify to get

$$\begin{aligned} T_{1:x} &= v^1(x) \left(\frac{1}{v^1(1)} T_{1:1} - \sum_{j=2}^{r} \frac{v^j(1)}{v^1(1)} M^j \right) + \left(\sum_{j=2}^{r} v^j(x) M^j \right) \\ &= \frac{v^1(x)}{v^1(1)} T_{1:1} + \sum_{j=2}^{r} \left(v^j(x) - \frac{v^j(1) v^1(x)}{v^1(1)} \right) M^j \end{aligned}$$

To find an expansion with $M^j = T_{1:j}$ for subsequent j, we simply repeat this procedure. We must be careful that we do not replace an earlier M^j. Fortunately the $T_{1:j}$ for $j \leq k$ are independent, so each one must use some M^j for $j > k$ when it is reached. ◀

We would like to extend this proof to delete rank 1 slices and simplify the tensor. With the same notation as in Lemma 4, write $T = \widetilde{T} + \sum_{h=1}^{k} v^h \otimes M^h$ and note that $\mathrm{rk}(T) = \mathrm{rk}(\widetilde{T}) + k$. We get the following consequence of Lemma 4.

▶ **Corollary 5.** *Given a d-tensor T of rank r, suppose the 1-slices $T_{1:j}$ for $j \in [k]$ are rank 1 and linearly independent as vectors. Then*

$$\mathrm{rk}(T) = k + \min(\mathrm{rk}(T - \sum_{j=1}^{k} v^j \otimes T_{1:j})),$$

where the minimum is taken over the choice of $\{v^j\}_{j \in [k]} \subseteq \mathbb{F}^{n_1}$. Furthermore, in the $T - \sum_{j=1}^{k} v^j \otimes T_{1:j}$ of minimum rank, we can assume slice $1 : j$, for $j \in [k]$, has all zero entries.

In Section 3 we use T' to refer to the minimum rank tensor in Corollary 5 with slices $1 : j$ removed, for $j \in [k]$.

3 Hardness

To establish NP-hardness of approximation we adapt the proof of NP-hardness by Håstad [6] but reduce from bounded occurrence SAT. Håstad started with a 3-SAT instance ϕ with n variables and m clauses and created a $(2+n+2m) \times 3n \times (3n+m)$ tensor T. If ϕ was satisfiable then $\mathrm{rk}(T) = 4n+2m$, and otherwise $\mathrm{rk}(T) > 4n+2m$. We follow the same general approach but by starting with a bounded occurence SAT instance we are able to more tightly relate the rank of the resulting tensor to the minimal number of falsified clauses.

3.1 Reduction

Let us give an overview of our reduction. Though the reduction works for a varying number of literals per clause, here we just reduce from MAX-E2-SAT. Given a SAT formula ϕ with n variables and m clauses, we create a $(1+n+m) \times 2n \times (n+m)$ tensor T with n variable slices and m clause slices. We represent each literal as a vector as follows: $v_{x_i} \in \mathbb{F}^{2n}$ contains a 1 in position $2i-1$ and 0 everywhere else. Vector $v_{\overline{x}_i} \in \mathbb{F}^{2n}$ contains a 1 in positions $2i-1$ and $2i$, and 0 everywhere else. The 3-slices are then defined as follows:

- For $i \leq n$, $T_{3:i}$ represents variable x_i. It has a 1 in positions $(1, 2i-1)$ and $(i+1, 2i)$, and is otherwise 0.
- Suppose the h^{th} clause is $C_h = \ell_1 \vee \ell_2$, where ℓ_i is a literal. Then
  - $(T_{3:(n+h)})_{1:1} = v_{\ell_1}$
  - $(T_{3:(n+h)})_{1:h+n+1} = v_{\ell_1} - v_{\ell_2}$
  - $T_{3:(n+h)}$ is 0 everywhere else

  Here are the slices for the formula $(x \vee y) \wedge (\overline{x} \vee \overline{y})$. We use "." to represent 0 for readability:

$$\begin{bmatrix} 1 & . & . & . \\ . & 1 & . & . \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \end{bmatrix}, \begin{bmatrix} . & . & 1 & . \\ . & . & . & . \\ . & . & . & 1 \\ . & . & . & . \\ . & . & . & . \end{bmatrix}, \begin{bmatrix} 1 & . & . & . \\ . & . & . & . \\ . & . & . & . \\ 1 & . & -1 & . \\ . & . & . & . \end{bmatrix}, \begin{bmatrix} 1 & 1 & . & . \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ 1 & 1 & -1 & -1 \end{bmatrix}$$

This tensor is a subset of the tensor from Håstad's paper [6], with all auxiliary 3-slices and the third copy of the variable columns removed. We use Corollary 5 to reduce the problem of computing the rank of T to that of computing a realization of some matrix M'. The matrix M' consists of:

- For each $i \in [n]$, a row v'_i, representing variable x_i. This row has a 1 in position $2i-1$, a variable a_i in position $2i$, and 0 everywhere else.
- For each $h \in [m]$, a row c'_h, representing clause $C_h = \ell_1 \vee \ell_2$. This row is nonzero only in the columns for variables appearing in the clause. It depends on a variable b_h and equals $(1-b_h)v_{\ell_1} + b_h v_{\ell_i}$.

  For example, for the formula $(x \vee y) \wedge (\overline{x} \vee \overline{y})$ we obtain

$$\begin{bmatrix} 1 & a_1 & . & . \\ . & . & 1 & a_2 \\ 1-b_1 & . & b_1 & . \\ 1-b_2 & 1-b_2 & b_2 & b_2 \end{bmatrix}, \text{ which might yield } \begin{bmatrix} 1 & . & . & . \\ . & . & 1 & 1 \\ 1 & . & . & . \\ . & . & 1 & 1 \end{bmatrix}$$

for a specific assignment to the variables. We use M to represent M' under some assignment to its variables, and call it a **realization** of M'. We use c_h to refer to c'_h and u_i to refer to v'_i under an assignment to the variables of M'. Then we have the following:

▶ **Lemma 6.**

$$\mathrm{rk}(T) = \min(\mathrm{rk}(M)) + n + m$$

where the minimum is taken over all realizations M of M'.

As one might expect, the variables in M' correspond to the choices taken in Corollary 5 of how to subtract slices.

Proof. We perform slice elimination on the 1-slices of T. We think of the third coordinate as being perpendicular to the page, so we can think of 1-slices as cutting horizontally across the page and also through the page. For $i \geq 2$ each 1-slice $T_{1:i}$ is a matrix with a single nonzero row. Only the 3-slice $T_{3:i-1}$ has nonzero values on $T_{1:i}$. Slice $T_{3:i-1}$ is the 3-slice for variable x_{i-1} if $i \leq n$ or clause C_{i-n-1} if $i > n$.

Each of these 1-slices thus has nonzero entries only in positions $T_{1:i}(x, i-1)$ for some x. The 1-slices are thus independent as vectors. We apply Corollary 5 and call the simplified tensor T'. There is a variable a_i in T' obtained from subtracting some multiple of $T_{1:i+1}$ from $T_{1:1}$ for $i \leq n$. There is a variable b_j obtained from subtracting some multiple of $T_{1:n+j+1}$ from $T_{1:1}$. Finally, T' has only size 1 in its 1^{st} coordinate, so we view it as the matrix $T'_{3:1} = M'$ and it has the same rank as T'. ◀

3.2 Bounding the rank

Now we use hardness properties of SAT. We define an *a-good* instance to be a formula ϕ with m clauses where some assignment leaves at most am clauses unsatisfied and an *a-bad* instance ψ to be a formula where every assignment leaves at least am clauses unsatisfied. We use the following theorem from Berman and Karpinski [2]:

▶ **Theorem 7.** *It is NP-hard to distinguish between $(4/792 + \epsilon)$-good and $(5/792 - \epsilon)$-bad instances of E3-OCC-MAX-2SAT, for any $\epsilon > 0$.*

We relate the rank of M to the fraction of clauses that ϕ can satisfy to obtain our main theorem.

▶ **Lemma 8** (Completeness). *If ϕ is a-good then for some realization M of M', we have $\mathrm{rk}(M) \leq n + am$.*

Proof. There is some assignment ρ so $\rho(C)$ is true for the maximum number of clauses $C \in \phi$. Set $a_i = 1 - \rho(x_i)$. Suppose clause $C_h = \ell_1 \vee \ell_2$, where ℓ_1 is a literal of x_i and ℓ_2 is a literal of x_j. If $\rho(\ell_1)$ is true then set $b_h = 0$ so that $c_h = v_{\ell_1} = u_i$. If instead $\rho(\ell_2)$ is true, then set $b_h = 1$ so that $c_h = v_{\ell_2} = u_j$. Thus no satisfied clause contributes to the rank of M. There are only am remaining rows, and hence $\mathrm{rk}(M) \leq n + am$. ◀

We conclude that the variable rows contribute n to the rank and satisfied clause rows contribute nothing. Unsatisfied clause rows contribute to the rank as well, but we have to be careful. In general it is hard to get a precise bound on the contribution of clauses to the rank of M. When many clauses share the same variable, we can only guarantee that the rank increases by some fraction of the number of unsatisfied clauses. However, for the very simple formulas we study, we can establish that Lemma 8 is sharp.

▶ **Lemma 9** (Soundness). *If ϕ is an a-bad instance of E3-OCC-MAX-2SAT then for every realization M of M', we have $\mathrm{rk}(M) \geq n + am$.*

Proof. Fix M, a realization of M' of minimum rank. We build a boolean assignment ρ that falsifies at most $\mathrm{rk}(M) - n$ clauses in ϕ. Say a row **uses** i if its last nonzero entry is in column $2i-1$ or $2i$. Sort the rows of M by the index used.

One row using i is u_i. There are at most 3 more rows using i, which we call r, s, and t, if they exist. Call the corresponding clauses C_r, C_s, and C_t. Let M_i be the submatrix of M consisting of those rows that use values up to i. If $\mathrm{rk}(M_i) - \mathrm{rk}(M_{i-1}) = 1$ then C_r, C_s, and C_t share the same literal of x_i. Call this literal ℓ_i. Then setting $\rho(\ell_i)$ true satisfies all these clauses. Otherwise, set $\rho(\ell_i)$ to satisfy the majority of the clauses C_r, C_s, and C_t.

Because there are only 3 clauses that contain x_i, this leaves at most 1 clause unsatisfied. There are at most $\mathrm{rk}(M) - n$ indices $i \in [n]$ where $\mathrm{rk}(M_i) - \mathrm{rk}(M_{i-1}) > 1$, so we end up with at most $\mathrm{rk}(M) - n$ falsified clauses. Since every assignment to ϕ falsifies at least am clauses, we conclude that $\mathrm{rk}(M) \geq n + am$. ◀

3.3 Inapproximability

Now we get our main theorem.

▶ **Theorem 10** (Theorem 1). *It is NP-hard to approximate 3-tensor rank over any field $\mathbb{F}$ within a factor of $1 + 1/1852 - \delta$, for any $\delta > 0$.*

Proof. We combine Theorem 7, Lemma 8, and Lemma 9. We know $\mathrm{rk}(T) = \min(\mathrm{rk}(M)) + n + m$. If ϕ is a-good then $\min(\mathrm{rk}(M)) \leq n + am$. If ϕ is a-bad then $\min(\mathrm{rk}(M)) \geq n + am$. By Theorem 7 it is NP-hard to distinguish $(4/792 + \epsilon)$-good and $(5/792 - \epsilon)$-bad instances. Hence, it is NP-hard to distinguish whether $\mathrm{rk}(T) \leq 2n + (1 + 4/792 + \epsilon)m$ or $\mathrm{rk}(T) \geq 2n + (1 + 5/792 - \epsilon)m$.

Since $m = \frac{3n}{2}$ in E3-OCC-MAX-2SAT, given $\delta > 0$ we can choose $\epsilon > 0$ so that we have an inapproximability ratio of $1 + 1/1852 - \delta$. ◀

4 Discussion

It seems likely that much better inapproximability results are possible for tensor rank. By counting, we know there are tensors of shape $n \times n \times n$ with rank at least $n^2/3$. Say that a family $T(n)$ of tensors of increasing size n is "explicit" if there is a polynomial time algorithm that takes input n written in unary and prints $T(n)$. Despite the existence of $n \times n \times n$ tensors of quadratic rank, the only "explicit" such tensors have rank at most $O(n)$ [15].

Alexeev et al. [1] claim that "any gap-preserving reduction from NP to tensor rank would automatically yield lower bounds for explicit tensors". While it is not obvious to us how to turn this into a technically precise statement, we sketch the main idea. Suppose we reduce SAT to 3-tensor rank and obtain a superconstant hardness of approximation for tensor rank. If the reduction always outputs a tensor T with largest dimension n, and where the slices in every direction are independent, then the tensor has rank at least n. If, for some $c > 3$, the reduction demonstrates c-hardness of approximation then it must sometimes output tensors of rank at least cn. If this happens on unsatisfiable formulas, we can apply the reduction to get an explicit high rank tensor.

While it is possible that a randomized reduction could avoid this barrier, we are still motivated to find high rank tensors. One approach to finding higher rank explicit 3-tensors or to improving the gap in a reduction is to Kronecker multiply together smaller 3-tensors. It is known [18] that the product is not multiplicative. For the curious reader, let us give a simple counterexample:

▶ **Example 11.** Take a 3-tensor T with slices

$$\begin{bmatrix} 1 & . \\ . & 1 \end{bmatrix} \begin{bmatrix} . & 1 \\ . & . \end{bmatrix}$$

Kronecker multiplying this with itself, we get a 3-tensor with slices

$$\begin{bmatrix} 1 & . & . & . \\ . & 1 & . & . \\ . & . & 1 & . \\ . & . & . & 1 \end{bmatrix} \begin{bmatrix} . & 1 & . & . \\ . & . & . & . \\ . & . & . & 1 \\ . & . & . & . \end{bmatrix} \begin{bmatrix} . & . & 1 & . \\ . & . & . & 1 \\ . & . & . & . \\ . & . & . & . \end{bmatrix} \begin{bmatrix} . & . & . & 1 \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \end{bmatrix}$$

The former has rank 3, but the latter set can be formed as a linear combination of 8 rank 1 matrices:

$$\begin{bmatrix} 1 & 1 & . & . \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \end{bmatrix} \begin{bmatrix} . & -1 & . & . \\ . & 1 & . & . \\ . & . & . & . \\ . & . & . & . \end{bmatrix} \begin{bmatrix} . & 1 & 1 & . \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \end{bmatrix} \begin{bmatrix} . & . & . & . \\ . & -1 & . & 1 \\ . & . & . & . \\ . & . & . & . \end{bmatrix} \begin{bmatrix} . & . & . & . \\ . & . & . & -1 \\ . & . & . & -1 \\ . & . & . & . \end{bmatrix}$$

$$\begin{bmatrix} . & . & . & 1 \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \end{bmatrix} \begin{bmatrix} . & . & . & . \\ . & . & . & . \\ . & . & 1 & . \\ . & . & . & . \end{bmatrix} \begin{bmatrix} . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ . & . & . & 1 \end{bmatrix}$$

T has rank 3, but when Kronecker multiplying with itself it has rank at most 8, not 9. This is true over any field $\mathbb{F}$.

Perhaps it is still true that when k copies of a 3-tensor T of shape $n \times n \times n$ with $\mathrm{rk}(T) > n$ are multiplied together, the rank grows as r^k for some $\mathrm{rk}(T) \geq r > n$. If true, this would immediately give high rank 3-tensors.

References

1. Boris Alexeev, Michael A. Forbes, and Jacob Tsimerman. Tensor rank: Some lower and upper bounds. *CoRR*, abs/1102.0072, 2011. `arXiv:1102.0072`.
2. Piotr Berman and Marek Karpinski. Efficient amplifiers and bounded degree optimization. *Electronic Colloquium on Computational Complexity (ECCC)*, 8(53), 2001. URL: `http://eccc.hpi-web.de/eccc-reports/2001/TR01-053/index.html`.
3. Markus Bläser. On the complexity of the multiplication of matrices of small formats. *J. Complexity*, 19(1):43–60, 2003. `doi:10.1016/S0885-064X(02)00007-9`.
4. Markus Bläser. Explicit tensors. In *Perspectives in Computational Complexity*, pages 117–130. Springer, 2014.
5. Markus Bläser, Christian Ikenmeyer, Gorav Jindal, and Vladimir Lysikov. Generalized matrix completion and algebraic natural proofs. *Electronic Colloquium on Computational Complexity (ECCC)*, 25:64, 2018. URL: `https://eccc.weizmann.ac.il/report/2018/064`.
6. Johan Håstad. Tensor rank is np-complete. *J. Algorithms*, 11(4):644–654, 1990. `doi:10.1016/0196-6774(90)90014-6`.
7. Tamara G. Kolda and Brett W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, 2009. `doi:10.1137/07070111X`.
8. Joseph B Kruskal. Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear algebra and its applications*, 18(2):95–138, 1977.

9 Julian D Laderman. A noncommutative algorithm for multiplying $3\times$ 3 matrices using 23 multiplications. *Bulletin of the American Mathematical Society*, 82(1):126–128, 1976.

10 J. M. Landsberg and Mateusz Michalek. Abelian tensors. *CoRR*, abs/1504.03732, 2015. `arXiv:1504.03732`.

11 Viñtor Yakovlevich Pan. Methods of computing values of polynomials. *Russian Mathematical Surveys*, 21(1):105–136, 1966.

12 Ran Raz. Tensor-rank and lower bounds for arithmetic formulas. *Electronic Colloquium on Computational Complexity (ECCC)*, 17:2, 2010. URL: `http://eccc.hpi-web.de/report/2010/002`.

13 Marcus Schaefer and Daniel Stefankovic. The complexity of tensor rank. *CoRR*, abs/1612.04338, 2016. `arXiv:1612.04338`.

14 Yaroslav Shitov. How hard is the tensor rank? *arXiv preprint arXiv:1611.01559*, 2016.

15 Amir Shpilka. Lower bounds for matrix product. *CoRR*, cs.CC/0201001, 2002. URL: `http://arxiv.org/abs/cs.CC/0201001`.

16 Zhao Song, David P. Woodruff, and Peilin Zhong. Relative error tensor low rank approximation. *CoRR*, abs/1704.08246, 2017. `arXiv:1704.08246`.

17 Volker Strassen. Gaussian elimination is not optimal. *Numerische mathematik*, 13(4):354–356, 1969.

18 Nengkun Yu, Eric Chitambar, Cheng Guo, and Runyao Duan. Tensor rank of the tripartite state $|w\rangle^{\otimes n}$. *Physical Review A*, 81(1):014301, 2010.

An $O(1)$-Approximation Algorithm for Dynamic Weighted Vertex Cover with Soft Capacity

Hao-Ting Wei
Department of Industrial Engineering and Engineering Management, National Tsing Hua University, Hsinchu 30013, Taiwan
s104034526@m104.nthu.edu.tw

Wing-Kai Hon
Department of Computer Science, National Tsing Hua University, Hsinchu 30013, Taiwan
wkhon@cs.nthu.edu.tw

Paul Horn[1]
Department of Mathematics, University of Denver, Denver, USA
paul.horn@du.edu

Chung-Shou Liao[2]
Department of Industrial Engineering and Engineering Management, National Tsing Hua University, Hsinchu 30013, Taiwan
csliao@ie.nthu.edu.tw

Kunihiko Sadakane
Department of Mathematical Informatics, The University of Tokyo, Tokyo, Japan
sada@mist.i.u-tokyo.ac.jp

Abstract

This study considers the *soft* capacitated vertex cover problem in a dynamic setting. This problem generalizes the dynamic model of the vertex cover problem, which has been intensively studied in recent years. Given a dynamically changing vertex-weighted graph $G = (V, E)$, which allows edge insertions and edge deletions, the goal is to design a data structure that maintains an approximate minimum vertex cover while satisfying the capacity constraint of each vertex. That is, when picking a copy of a vertex v in the cover, the number of v's incident edges covered by the copy is up to a given capacity of v. We extend Bhattacharya et al.'s work [SODA'15 and ICALP'15] to obtain a deterministic primal-dual algorithm for maintaining a constant-factor approximate minimum capacitated vertex cover with $O(\log n/\epsilon)$ amortized update time, where n is the number of vertices in the graph. The algorithm can be extended to (1) a more general model in which each edge is associated with a non-uniform and unsplittable demand, and (2) the more general capacitated set cover problem.

2012 ACM Subject Classification Theory of computation → Dynamic graph algorithms

Keywords and phrases approximation algorithm, dynamic algorithm, primal-dual, vertex cover

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.27

Related Version https://arxiv.org/abs/1802.05623

[1] Supported by NSA Young Investigator Grant H98230-15-1-0258, and Simons Collaboration Grant #525039.
[2] Supported by MOST Taiwan under Grants MOST105-2628-E-007-010-MY3 and MOST105-2221-E-007-085-MY3.

1 Introduction

Dynamic algorithms have received fast-growing attention in the past decades, especially for some classical combinatorial optimization problems such as connectivity [1, 8, 11], vertex cover, and maximum matching [2, 3, 4, 5, 13, 14, 15, 16]. This paper focuses on the fully dynamic model of the vertex cover problem, which has been intensively studied in recent years. Given a vertex-weighted graph $G = (V, E)$ which is constantly updated due to a sequence of edge insertions and edge deletions, the objective is to maintain a subset of vertices $S \subseteq V$ at any given time, such that every edge is incident to at least one vertex in S and the weighted sum of S is minimized. We consider a generalization of the problem, where each vertex is associated with a given capacity. When picking a copy of a vertex v in S, the number of its incident edges that can be covered by such a copy is bounded by v's given capacity. The objective is to find a *soft* capacitated weighted vertex cover S with minimum weight, i.e. $\sum_{v \in S} c_v x_v$ is minimized, as well as an assignment of edges such that the number of edges assigned to a vertex v in S is at most $k_v x_v$, where c_v is the cost of v, k_v is the capacity of v, and x_v is the number of selected copies of v in S. Assume there is no bound on x_v. The static model of this generalization is the so-called *soft capacitated vertex cover* problem, introduced by Guha et al. [9]. [3]

Prior work. For the vertex cover problem in a dynamic setting, Ivkovic and Lloyd [12] presented the pioneering work wherein their fully dynamic algorithm maintains a 2-approximation factor to vertex cover with $O((n+m)^{0.7072})$ update time, where n is the number of vertices and m is the number of edges. Onak and Rubinfeld [14] designed a randomized data structure that maintains a large constant approximation ratio with $O(\log^2 n)$ amortized update time in expectation; this is the first result that achieves a constant approximation factor with polylogarithmic update time. Baswana, Gupta, and Sen [2] designed another randomized data structure which improves the approximation ratio to two, and simultaneously improved the amortized update time to $O(\log n)$. Recently, Solomon [16] gave the currently best randomized algorithm, which maintains a 2-approximate vertex cover with $O(1)$ amortized update time.

For deterministic data structures, Onak and Rubinfeld [14] presented a data structure that maintains an $O(\log n)$-approximation algorithm with $O(\log^2 n)$ amortized update time. Bhattacharya et al. [5] proposed the first deterministic data structure that maintains a constant ratio, precisely, a $(2+\epsilon)$-approximation to vertex cover with polylogarithmic $O(\log n/\epsilon^2)$ amortized updated time. Existing work also considered the worst-case update time. Neiman and Solomon [13] provided a 2-approximation dynamic algorithm with $O(\sqrt{m})$ worst-case update time. Later, Peleg and Solomon [15] improved the worst-case update time to $O(\gamma/\epsilon^2)$, where γ is the arboricity of the input graph. Very recently, Bhattacharya et al. [3] extended their hierarchical data structure to achieve the currently best worst-case update time of $O(\log^3 n)$. Note that the above studies only discussed the unweighted vertex cover problem, the objective of which is to find a vertex cover with minimum cardinality.

Consider the dynamic (weighted) set cover problem. Bhattacharya et al. [6] used a hierarchical data structure similar to that reported in [5], and achieved a scheme with $O(f^2)$-approximation ratio and $O(f \log(n+m)/\epsilon^2)$ amortized updated time, where f is the maximum frequency of an element. Very recently, Gupta et al. [10] improved the amortized

[3] If each x_v is associated with a bound, it is called the *hard capacitated vertex cover* problem, introduced by Chuzhoy and Naor [7].

Table 1 Summary of results for unweighted (resp. weighted) minimum vertex cover (UMVC (resp. WMVC)), unweighted (resp. weighted) minimum set cover (UMSC (resp. WMSC)), where f is the maximum frequency of an element, and weighted minimum capacitated vertex (resp. set) cover (WMCVC (resp. WMCSC)).

Problem	**Approx. Guarantee**	Update Time	**Data Structure**	Reference
UMVC	$O(1)$	$O(\log^2 n)$ amortized	randomized	STOC'10 [14]
UMVC	2	$O(\log n)$ amortized	randomized	FOCS'11 [2]
UMVC	2	$O(1)$ amortized	randomized	FOCS'16 [16]
UMVC	2	$O(\sqrt{m})$ worst-case	deterministic	STOC'13 [13]
UMVC	$2+\epsilon$	$O(\log n/\epsilon^2)$ amortized	deterministic	SODA'15 [5]
UMVC	$2+\epsilon$	$O(\gamma/\epsilon^2)$ worst-case	deterministic	SODA'16 [15]
UMVC	$2+\epsilon$	$O(\log^3 n)$ worst-case	deterministic	SODA'17 [3]
WMVC	$2+\epsilon$	$O(\log n/\epsilon^2)$ amortized	deterministic	This paper
UMSC	$O(f^3)$	$O(f^2)$ amortized	deterministic	IPCO'17 [4]
WMSC	$O(f^2)$	$O(f\log(n+m)/\epsilon^2)$ amortized	deterministic	ICALP'15 [6]
WMSC	$O(f^3)$ $O(\log n)$	$O(f^2)$ amortized $O(f\log n)$ amortized	deterministic	STOC'17 [10]
WMCVC	$O(1)$	$O(\log n/\epsilon)$ amortized	deterministic	This paper
WMCSC	$O(f^2)$	$O(f\log(n+m)/\epsilon)$ amortized	deterministic	This paper

update time to $O(f^2)$, albeit the dynamic algorithm achieves a higher approximation ratio of $O(f^3)$. They also offered another $O(\log n)$-approximation dynamic algorithm in $O(f\log n)$ amortized update time. Bhattacharya et al. [4] simultaneously derived the same outcome with $O(f^3)$-approximation ratio and $O(f^2)$ amortized update time for the unweighted set cover problem. Table 1 presents a summary of the above results.

Our contribution. In this study we investigate the *soft* capacitated vertex cover problem in the dynamic setting, where there is no bound on the number of copies of each vertex that can be selected. We refer to the primal-dual technique reported in [9], and present the first deterministic algorithm for this problem, which can maintain an $O(1)$-approximate minimum capacitated (weighted) vertex cover with $O(\log n/\epsilon)$ amortized update time. The algorithm can be extended to a more general model in which each edge is associated with a given demand, and the demand has to be assigned to an incident vertex. That is, the demand of each edge is *nonuniform* and *unsplittable*. Also, it can be extended to solve the more general capacitated set cover problem, where the input graph is a hyper-graph, and each edge may connect to multiple vertices.

The proposed dynamic mechanism builds on Bhattacharya et al.'s (α,β)-partition structure [5, 6], but a *careful adaptation* has to be made to cope with the newly introduced capacity constraint. Briefly, applying the *fractional matching* technique in Bhattacharya et al.'s algorithm cannot directly lead to a constant approximation ratio in the capacitated vertex cover problem. The crux of our result is the re-design of a key parameter, *weight* of a vertex, in the dual model. Details of this modification are shown in the next section.

In addition, if we go back to the original vertex cover problem without capacity constraint, the proposed algorithm is able to resolve the *weighted* vertex cover problem by maintaining a $(2+\epsilon)$-approximate weighted vertex cover with $O(\log n/\epsilon^2)$ amortized update time. This result achieves the same approximation ratio as the algorithm in [5], but they considered the unweighted model. Details of this discussion are presented in the end of Section 3.

1.1 Overview of our technique

First, we recall the mathematical model of the capacitated vertex cover problem which was first introduced by Guha et al. [9]. In this model, y_{ev} serves as a binary variable that indicates whether an edge e is covered by a vertex v. Let N_v be the set of incident edges of v, k_v and c_v be the capacity and the cost of a vertex v, respectively. Let x_v be the number of selected copies of a vertex v. An integer program (IP) model of the problem can be formulated as follows (the minimization program on the left):

Min	$\sum_v c_v x_v$		**Max**	$\sum_{e\in E} \pi_e$	
s.t	$y_{ev} + y_{eu} \geq 1,$	$\forall e = \{u, v\} \in E$	**s.t**	$k_v q_v + \sum_{e\in N_v} l_{ev} \leq c_v,$	$\forall v \in V$
	$k_v x_v - \sum_{e\in N_v} y_{ev} \geq 0,$	$\forall v \in V$		$q_v + l_{ev} \geq \pi_e,$	$\forall v \in e, \forall e \in E$
	$x_v \geq y_{ev},$	$\forall v \in e, \forall e \in E$		$q_v \geq 0,$	$\forall v \in V$
	$y_{ev} \in \{0, 1\},$	$\forall v \in e, \forall e \in E$		$l_{ev} \geq 0,$	$\forall v \in e, \forall e \in E$
	$x_v \in \mathbb{N},$	$\forall v \in V$		$\pi_e \geq 0,$	$\forall e \in E$

If we allow a relaxation of the above primal form, i.e., dropping the integrality constraints, its dual problem yields a maximization problem. The linear program for the dual can be formulated as shown in the above (the maximization program on the right; also see [9]). One may consider this as a variant of the *packing* problem, where we want to pack a value of π_e for each edge e, so that the sum of the packed values is maximized. Packing of e is limited by the sum of q_v and l_{ev}, where q_v is the *global* ability of a vertex v emitted to v's incident edges, and l_{ev} is the *local* ability of v distributed to its incident edge e.

In this study, we incorporate the above IP model with its LP relaxation for capacitated vertex cover into the dynamic mechanism proposed by Bhattacharya et al. [5, 6]. They devised the *weight* of a vertex v (in the dual model), denoted by W_v, to obtain a feasible solution in the dual problem. They also allowed a flexible range for W_v to quickly adjust the solution for dynamic updates while preserving its approximation quality. Due to the additional capacity constraint in our problem, a new *weight* function is obviously required.

Technical challenges. There are two major differences between our algorithm and Bhattacharya et al.'s [5, 6]. First, the capacity constraint in the primal problem leads to the two variables q_v and l_{ev} in the dual problem in which we have to balance their values when approaching c_v to maximize the dual objective. By contrast, the previous work considered one dual variable l_{ev} without the restriction on the *coverage* of a vertex. We thus re-design W_v, the *weight* of a vertex v to specifically consider the capacitated scenario. Yet, even with the new definition of W_v, there is still a second challenge on how to approximate the solution within a constant factor in the dynamic environment. In order to achieve $O(\log n)$ amortized update time, Bhattacharya et al.'s *fractional matching* approach assigns the value of all v's incident edges to v, which, however, may result in a non-constant h, hidden in the approximation ratio, where h is the largest number of copies selected in the cover. We observe that we cannot remove h from the approximation guarantee based on the (α, β)-partition structure if we just select the minimum value of α, as it is done in [5, 6]. The key insight is that we show a bound on the value of α, which restricts the updates of the dynamic mechanism. With the help of this insight, we are able to revise the setting of α to derive a constant approximation ratio, while maintaining the $O(\log n)$ update time.

2 Level Scheme and its Key Property

The core of Bhattacharya et al.'s (α, β)-partition structure [5, 6] is a *level scheme* [14] that is used to maintain a feasible solution in their dual problem. In this section, we demonstrate

(in a different way from the original papers) how this scheme can be applied to our dual problem, and describe the key property that the scheme guarantees.

A *level scheme* is an assignment $\ell : V \to \{0, 1, \ldots, L\}$ such that every vertex $v \in V$ has a level $\ell(v)$. Let $c_{\min}$ and $c_{\max}$ denote the minimum and maximum costs of a vertex, respectively. For our case, we set $L = \lceil \log_\beta(n\mu\alpha/c_{\min}) \rceil$ for some $\alpha, \beta > 1$ and $\mu > c_{\max}$. Based on ℓ, each edge (u, v) is also associated with a level $\ell(u, v)$, where $\ell(u, v) = \max\{\ell(u), \ell(v)\}$. An edge is assigned to the higher-level endpoint, and ties are broken arbitrarily if both endpoints have the same level.

Each edge (u, v) has a weight $w(u, v)$ according to its level, such that $w(u, v) = \mu\beta^{-\ell(u,v)}$. Each vertex v also has a weight W_v, which is defined based on the incident edges of v and their corresponding levels. Before giving details on W_v, we first define some notations. Let $N_v = \{u \mid (u, v) \in E\}$ be the set of vertices adjacent to v (i.e., the neighbors of v). Let $N_v(i)$ denote the set of level-i neighbors of v, and $N_v(i, j)$ denote the set of v's neighbors whose levels are in the range $[i, j]$. That is, $N_v(i) = \{u \mid (u, v) \in E \wedge \ell(u) = i\}$ and $N_v(i, j) = \{u \mid (u, v) \in E \wedge \ell(u) \in [i, j]\}$. The *degree* of a vertex v is denoted by $D_v = |N_v|$. Similarly, we define $D_v(i) = |N_v(i)|$ and $D_v(i, j) = |N_v(i, j)|$. Finally, we use $\delta(v)$ to denote the set of edges assigned to a vertex v. Now, the weight W_v of a vertex v is defined as follows:

Case 1 $D_v(0, \ell(v)) > k_v$:

$$W_v = k_v\mu\beta^{-\ell(v)} + \sum_{i>\ell(v)} \min\{k_v, D_v(i)\}\mu\beta^{-i}$$

Case 2 $D_v(0, \ell(v)) \le k_v$:

$$W_v = D_v(0, \ell(v))\mu\beta^{-\ell(v)} + \sum_{i>\ell(v)} \min\{k_v, D_v(i)\}\mu\beta^{-i}$$

Due to the capacity constraint, we consider whether the number of level-i neighbors of v, $0 \le i \le \ell(v)$, is larger than the capacity of v, to define the weight of a vertex v. Note that the total weight of the edges that are assigned to v or incident to v can contribute at most $k_v w(u, v)$ to W_v. Briefly, the weight of a vertex has two components: one that is dependent on the incident edges with level $\ell(v)$, and the other that is dependent on the remaining incident edges. For convenience, we call the former component $Internal_v$ and the latter component as $External_v$. Moreover, we have:

$$External_v \quad \le \quad k_v \sum_{i>\ell(v)} \mu\beta^{-i} \quad \le \quad (1/(\beta - 1))k_v\mu\beta^{-\ell(v)}.$$

In general, an arbitrary level scheme cannot be used to solve our problem. What we need is a *valid* level scheme, which is defined as follows.

▶ **Definition 1.** A level scheme is *valid* if $W_v \le c_v$, for every vertex v.

▶ **Lemma 2.** *Let V_0 denote the set of level-0 vertices in a valid level scheme. Then, $V \setminus V_0$ forms a* vertex cover *of G.*

Proof. Consider any edge $(u, v) \in E$. We claim that at least one of its endpoints must be in $V \setminus V_0$. Suppose that the claim is false which implies that $\ell(u) = \ell(v) = 0$ and $w(u, v) = \mu > c_{\max}$. Since $w(u, v)$ appears in $Internal_v$, we have $W_v \ge w(u, v)$. As a result, $c_v \ge W_v \ge \mu > c_{\max}$, which leads to a contraditction. The claim thus follows, and so does the lemma. ◀

The above lemma implies that no edge is assigned to any level-0 vertex. In our mechanism, we will maintain a valid level scheme, based on which each vertex in $V \setminus V_0$ picks enough copies to cover all the edges assigned to it; this forms a valid capacitated vertex cover.

Next, we define the notion of *tightness*, which is used to measure how good a valid level scheme performs.

▶ **Definition 3.** A valid level scheme with an associated edge assignment is *ε-tight* if for every vertex v with $|\delta(v)| > 0$, $W_v \in (c_v/\varepsilon, c_v]$.

▶ **Lemma 4.** *Given an ε-tight valid level scheme, we can obtain an $\varepsilon(2(\beta/(\beta-1))+1)$-approximation solution to the* weighted minimum capacitated vertex cover (WMCVC) *problem.*

Proof. First, we fix an arbitrary edge assignment that is consistent with the given valid level scheme. For each vertex v with $|\delta(v)| > 0$, we pick $\lceil |\delta(v)|/k_v \rceil$ copies to cover all the $|\delta(v)|$ edges assigned to it. To analyze the total cost of this capacitated vertex cover, we relate it to the value $\sum_e \pi_e$ of a certain feasible solution of the dual problem, whose corresponding values of q_v and l_{ev} are as follows:

For every vertex v:

- if $\lceil |\delta(v)|/k_v \rceil > 1$: $q_v = \mu\beta^{-\ell(v)}$, and $l_{ev} = 0$;
- if $\lceil |\delta(v)|/k_v \rceil \le 1$: $q_v = \mu \sum_{i | D_v(i) > k_v} \beta^{-i}$, $l_{ev} = 0$ if $D_v(\ell(e)) > k_v$, and $l_{ev} = \mu\beta^{-\ell(e)}$ otherwise.

For every edge e: $\pi_e = \mu\beta^{-\ell(e)}$.

It is easy to verify that the above choices of q_v, l_{ev}, and π_e give a feasible solution to the dual problem.

For the total cost of our solution, we separate the analysis into two parts, based on the multiplicity of the vertex:

Case 1 $\lceil |\delta(v)|/k_v \rceil > 1$: In this case, the external component of W_v is at most $1/(\beta-1)$ of the internal component, so $W_v \le (\beta/(\beta-1))k_v q_v$. Then, the cost of all copies of v is:

$$\begin{aligned}\lceil |\delta(v)|/k_v \rceil \cdot c_v &\le \lceil |\delta(v)|/k_v \rceil \cdot \varepsilon \cdot W_v \\ &\le 2 \cdot \frac{|\delta(v)|}{k_v} \cdot \varepsilon \cdot (\beta/(\beta-1))k_v q_v \;=\; 2\varepsilon(\beta/(\beta-1)) \cdot \sum_{e \in \delta(v)} \pi_e.\end{aligned}$$

Case 2 $\lceil |\delta(v)|/k_v \rceil = 1$: In this case, we pick one copy of vertex v, whose cost is:

$$c_v \le \varepsilon \cdot W_v \;\le \varepsilon \cdot \sum_{e \sim v} \pi_e \;=\; \varepsilon \cdot \left(\sum_{e \in \delta(v)} \pi_e + \sum_{e \notin \delta(v),\, e \sim v} \pi_e \right),$$

where $e \sim v$ denotes e is an edge incident to v.

In summary, the total cost is bounded by

$$\begin{aligned}&\sum_v \left(\max\{\varepsilon, 2\varepsilon(\beta/(\beta-1))\} \sum_{e \in \delta(v)} \pi_e + \varepsilon \sum_{e \notin \delta(v),\, e \sim v} \pi_e \right) \\ &= \sum_v \left(2\varepsilon(\beta/(\beta-1)) \sum_{e \in \delta(v)} \pi_e + \varepsilon \sum_{e \notin \delta(v),\, e \sim v} \pi_e \right) \\ &= \varepsilon(2(\beta/(\beta-1))+1) \sum_e \pi_e \\ &\le \varepsilon(2(\beta/(\beta-1))+1) \cdot OPT,\end{aligned}$$

where OPT denotes the optimal solution of the dual problem, which is also a lower bound of the cost of any weighted capacitated vertex cover. ◀

The next section discusses how to dynamically maintain an ε-tight level scheme, for some constant factor ε and with amortized $O(\log n/\epsilon)$ update time. Before that, we show a greedy approach to get a $(\beta+1)$-tight level scheme to the static problem as a warm up.

First, we have the following definition.

▶ **Definition 5.** A valid level scheme λ is *improvable* if some vertex can drop its level to get another level scheme λ' such that λ' is valid; otherwise, we say λ is *non-improvable*.

▶ **Lemma 6.** *If a valid level scheme λ is non-improvable, then λ is $(\beta+1)$-tight.*

If we set the level of every vertex to L initially, it is easy to check that by our choice of L as $\lceil \log_\beta(n\mu\alpha/c_{\min})\rceil$, such a level scheme is valid. Next, we examine each vertex one by one, and drop its level as much as possible while the scheme remains valid. In the end, we will obtain a non-improvable scheme, so that by the above lemma, the scheme is $(\beta+1)$-tight. This implies a $(\beta+1)(2(\beta/(\beta-1))+1)$-approximate solution for the WMCVC problem.

3 Maintaining an $\alpha(\beta+1)$-tight Level Scheme Dynamically

In this section, we present our $O(1)$-approximation algorithm for the WMCVC problem, with amortized $O(\log n)$ update time for each edge insertion and edge deletion. We first state an invariant that is maintained throughout by our algorithm, and show how the latter is done. Next, we analyze the time required to maintain the invariant with the potential method, and show that our proposed method can be updated efficiently as desired. To obtain an $O(\log n)$ amortized update time, we relax the flexible range of the weight of a vertex W_v by multiply a constant α. Let c^*_v be $c_v/\alpha(\beta+1)$. The invariant that we maintain is as follows.

▶ **Invariant 7.** *(1) For every vertex $v \in V \setminus V_0$, it holds that $c^*_v \le W_v \le c_v$, and (2) for every vertex $v \in V_0$, it holds that $W_v \le c_v$.*

By maintaining the above invariant, we will automatically obtain an $\alpha(\beta+1)$-tight valid scheme. As mentioned, we will choose a value for α in order to remove h from the approximation ratio. In particular, we will set $\alpha = (2\beta+1)/\beta + 2\epsilon$, where $0 < \epsilon < 1$ to balance the update time, and $\beta = 2.43$ to minimize the approximation ratio, so that we achieve the following theorem.

▶ **Theorem 8.** *There exists a dynamic level scheme λ which can achieve a constant approximation ratio (≈ 36) for the WMCVC problem with $O(\log n/\epsilon)$ amortized update time.*

The remainder of this section is devoted to proving Theorem 8.

3.1 The algorithm: Handling insertion or deletion of an edge

We now show how to maintain the invariant under edge insertions and deletions. A vertex is called *dirty* if it violates Invariant 7, and *clean* otherwise. Initially, the graph is empty, so that every vertex is clean and is at level zero. Assume that at the time instant just prior to the t^{th} update, all vertices are clean. When the t^{th} update takes place, which either inserts or deletes an edge $e = (u, v)$, we need to adjust the weights of u and v accordingly. Due to this adjustment, the vertices u, or v, or both may become dirty. To recover from this, we call the procedure Fix. The pseudo codes of the update algorithm (Algorithm 1) and the procedure Fix are shown in the next page.

Algorithm 1

if an edge $e = (u, v)$ has been inserted **then**
    Set $\ell(e) = \max\{\ell(u), \ell(v)\}$ and set $w(u, v) = \mu\beta^{-\ell(e)}$
    Update W_u and W_v
else if an edge $e = (u, v)$ has been deleted **then**
    Update W_u and W_v
end if
Run procedure FIX

procedure FIX:

while there exists a dirty vertex v **do**
    if $W_v > c_v$ **then**
        Increment the level of v by setting $\ell(v) \leftarrow \ell(v) + 1$
        Update W_v and W_u for all affected v's neighboring vertices u
    else if $W_v < c_v^*$ and $\ell(v) > 0$ **then**
        Decrement the level of v by setting $\ell(v) \leftarrow \ell(v) - 1$
        Update W_v and W_u for all affected v's neighboring vertices u
    end if
end while

Algorithm 1 ensures that Invariant 7 is maintained after each update, so that the dynamic scheme is $\alpha(\beta + 1)$-tight as desired. To complete the discussion, as well as the proof of Theorem 8, it remains to show that each update can be performed efficiently, in amortized $O(\log n)$ time.

3.2 Time complexity

Each update involves two steps, namely the adjustment of weights of the endpoints, and the running of procedure FIX. We now give the time complexity analysis, where the main idea is to prove the following two facts: **(Fact 1)** the amortized cost of the adjustment step is $O(\log n)$, and **(Fact 2)** the amortized cost of the procedure FIX is zero, irrespective of the number of vertices or edges that are affected during this step. Once the above two facts are proven, the time complexity analysis follows.

We use the standard potential method in our amortized analysis. Imagine that we have a bank account B. Initially, the graph is empty, and the bank account B has no money. For each adjustment step during an edge insertion or deletion, we deposit some money into the bank account B; after that, we use the money in B to pay for the cost of the procedure FIX. Some proofs are omitted in the following due to space limit.

Following the definition of [6], we say a vertex $v \in V$ is *active* if its degree in G is non-zero, and *passive* otherwise. Now, the value of B is set by the following formula:

$$B \;=\; \frac{1}{\epsilon} \cdot \left(\sum_{e \in E} \phi(e) + \sum_{v \in V} \psi(v) \right),$$

where $0 < \epsilon < 1$, and ϕ and ψ are functions defined as follows:

$$\phi(e) = \left(\frac{\beta}{(\beta - 1)} + \epsilon \right) (L - \ell(e)).$$

$$\psi(v) = \begin{cases} \frac{\beta^{(\ell(v)+1)}}{\mu(\beta-1)} \cdot \max\{0, \alpha\, c_v^* - W_v\}, & \text{if } v \text{ is } \textit{active}. \\ 0, & \text{otherwise.} \end{cases}$$

The following lemma proves Fact 1.

▶ **Lemma 9.** *After the adjustment step, the potential B increases by at most $O(\log n/\epsilon)$.*

We now switch our attention to Fact 2. Observe that the procedure Fix performs a series of level up and level down events. For each such event, the level of a specific vertex v will be changed, which will then incur a change in its weight, and changes in the weights of some of the incident edges and their endpoints. Let t_0 denote the moment before a level up or a level down event, and t_1 denote the moment after the weights of the edges and vertices are updated due to this event. Let Count denote the number of times an edge in the graph G is updated (for simplicity, we assume that in one edge update, the weight and the assignment of the edge may be updated, and so do the weights of its endpoints, where all these can be done in $O(1)$ time).

For ease of notation, in the following, a superscript t in a variable denotes the variable at moment t. For instance, $W_v^{t_0}$ stands for the weight W_v of v at moment t_0. Also, we use Δx to denote the quantity $x^{t_0} - x^{t_1}$, so that

$$|\Delta\textsc{Count}| = |\textsc{Count}^{t_0} - \textsc{Count}^{t_1}| = \textsc{Count}^{t_1} - \textsc{Count}^{t_0}$$

represents the number of incident edges whose weights are changed between t_0 and t_1.

Briefly speaking, based on the level scheme and the potential function B, we can show:

- For each level up event, each of the affected edges e would have its $\phi(e)$ value dropped, so that an ϵ fraction can pay for the weight updates of itself and its endpoints, while the remaining fraction can be converted into the increase in $\psi(v)$ value.
- For each level down event, the reverse happens, where the vertex v would have its $\psi(v)$ value dropped, so that an ϵ fraction can pay for the weight updates of the affected edges and their endpoints, while the remaining fraction can be converted into the increase in $\phi(e)$ values of the affected edges. The α value controls the frequency of the level down events, while trading this off with the approximation guarantee.

Sections 3.2.1 and 3.2.2 present the details of the amortized analysis of these two types of events, respectively. Finally, note that there is no money (potential) input to the bank B after the adjustment step, so that the analysis implies that the procedure Fix must stop (as the money in the bank is finite).

3.2.1 Amortized cost of level up

Let v be the vertex that undergoes the level up event, and $i = \ell(v)$ denote its level at moment t_0. By our notation, $\Delta B = B^{t_0} - B^{t_1}$ denotes the potential *drop* in the bank B from moment t_0 to moment t_1. To show that the amortized cost of a level up event is at most zero, it is equivalent to show that $\Delta B \geq |\Delta\textsc{Count}|$.

Recall that after a level up event, only the value of $\psi(v)$, the values of $\phi(e)$ and $\psi(u)$ for an edge (u, v) may be affected. In the following, we will examine carefully the changes in such values, and derive the desired bound for ΔB. First, we have the following simple lemma.

▶ **Lemma 10.** $|\Delta\textsc{Count}| \leq D_v^{t_0}(0, i)$.

Proof. When v changes from level i to $i+1$, only those incident edges with levels i will be affected. ◀

The next three lemmas examine, respectively, the changes $\Delta\psi(v)$, $\Delta\phi(e)$, and $\Delta\psi(u)$.

▶ **Lemma 11.** $\Delta\psi(v) = 0$.

▶ **Lemma 12.** *For every edge e incident to v,*

$$\Delta\phi(e) = \begin{cases} \left(\frac{\beta}{(\beta-1)} + \epsilon\right), & \text{if } \ell(e) \in [0, i]. \\ 0, & \text{otherwise.} \end{cases}$$

▶ **Lemma 13.** *For every vertex $u \in N_v^{t_0}$, $\Delta\psi(u) \geq -\beta/(\beta-1)$.*

Based on the above lemmas, we derive the following and finish the proof for the case of level up.

$$\begin{aligned} \Delta B &= \frac{1}{\epsilon} \cdot \left(\Delta\psi(v) + \sum_{e \in E} \Delta\phi(e) + \sum_{u \in N_v^{t_0}} \Delta\psi(u) \right) \\ &\geq \frac{1}{\epsilon} \cdot \left(0 + \left(\frac{\beta}{(\beta-1)} + \epsilon\right) D_v^{t_0}(0,i) - \frac{\beta}{\beta-1} D_v^{t_0}(0,i) \right) \\ &= D_v^{t_0}(0,i) \quad \geq \quad |\Delta\textsc{Count}|. \end{aligned}$$

3.2.2 Amortized cost of level down

We now show that the amortized cost of level down for a vertex v is at most zero. Similar to the case of level up, we examine $\Delta\psi(v)$, $\Delta\phi(e)$, and $\Delta\psi(u)$, and show that $\Delta B \geq |\Delta\textsc{Count}|$.

Before starting the proof of the level down case, recall that we have mentioned a parameter h at the end of Introduction, where h is the largest number of selected copies of all the vertices. That is, $h = \max_v\{\lceil |\delta^{t_0}(v)|/k_v \rceil\}$. Also, we let $h' = \max_v\{\lceil D_v^{t_0}(0, \ell(v))/k_v \rceil\}$, where $h' \geq h$, and set $\xi \geq 0$ such that $h' = h + \xi$.

▶ **Lemma 14.** $|\Delta\textsc{Count}| \leq D_v^{t_0}(0,i) < h' \cdot \frac{\beta^i c_v^*}{\mu}$.

Now, we are ready to examine $\Delta\psi(v)$, $\Delta\phi(e)$, and $\Delta\psi(u)$, through the following lemmas.

▶ **Lemma 15.** *For every vertex $u \in N_v^{t_0}$, $\Delta\psi(u) \geq -1/(\beta-1)$.*

Next, we partition $N_v^{t_0}$ into three subsets: X, Y_1 and Y_2, i.e. $N_v^{t_0} = X \cup Y_1 \cup Y_2$, where

$$\begin{aligned} X &= \{u \mid u \in N_v^{t_0}(0, i-1)\}, \\ Y_1 &= \{u \mid u \in N_v^{t_0}(i)\}, \\ Y_2 &= \{u \mid u \in N_v^{t_0}(i+1, L)\}. \end{aligned}$$

▶ **Lemma 16.** *For every edge (u,v) incidents to a vertex v,*

$$\Delta\phi(u,v) = \begin{cases} -\left(\frac{\beta}{(\beta-1)} + \epsilon\right), & \text{if } u \in X \\ 0, & \text{if } u \in Y_1 \cup Y_2. \end{cases}$$

Next, let $W_v^{t_0} = x + y_1 + y_2$, where x, y_1 and y_2 on the right-hand-side correspond to the weights generated by the subsets X, Y_1, Y_2, respectively. So, we get the following lemmas:

▶ **Lemma 17.** $\sum_{u \in N_v^{t_0}} \Delta\phi(u,v) \leq -\left(\frac{\beta}{(\beta-1)} + \epsilon\right)(hx\beta^i/\mu)$.

▶ **Lemma 18.** $\Delta\psi(v) = (\alpha c_v^* - x - y_1 - y_2) \cdot \frac{\beta^{i+1}}{\mu(\beta-1)} - \max\{0, \alpha c_v^* - \beta x - y_1 - y_2\} \cdot \frac{\beta^i}{\mu(\beta-1)}$.

Finally, depending upon the value of $\alpha c_v^* - \beta x - y_1 - y_2$, we consider two possible scenarios, where we show that in each case, $\Delta B \geq h' \cdot \beta^i c_v^*/\mu$. This in turn implies $\Delta B \geq |\Delta\textsc{Count}|$ as desired. Thus, the level scheme remains $\alpha(\beta+1)$-tight after a level down event. However, the value of h is bounded by n, and h appears inside α, so that the approximation ratio of the scheme may become n in the worst-case. Fortunately, with the help of the following lemma, we can choose α carefully, which in turn improves the approximation ratio from n to $O(1)$.

▶ **Lemma 19.** *Suppose that we set $\alpha \geq \beta/(\beta-1)$. By the time a level down event occurs at v at moment t_0, exactly one copy of v is selected. That is, $\lceil |\delta^{t_0}(v)|/k_v \rceil = 1$.*

Proof. Assume to the contrary that v could decrease its level even if more than one copy of v is selected. Since v undergoes level down, its weight W_v must have decreased; this can happen only in one of the following cases:

Case 1: An incident edge whose level is in the range $[0, \ell(v)]$ is deleted. In this case, since more than one copy of v is selected, W_v is unchanged. Thus, this case cannot happen.

Case 2: An incident edge whose level is in the range $[\ell(v)+1, L]$ is deleted. In this case, the weight $W_v^{t_0}$ at moment t_0 is less than c_v^*. On the other hand, at the moment t' when v attains the current level $\ell(v)$ (from level $\ell(v)-1$), its weight $W_v^{t'}$ was at least c_v before level up, and became at least $c_v/(\beta+1)$ after the level up. (The reason is from the proof of Lemma 6: the weight change between consecutive levels is at most a factor of $\beta+1$.) This implies that:

$$\begin{aligned} c_v^* \;&>\; W_v^{t_0} \;\geq\; k_v\mu\beta^{-\ell(v)} \;\because \text{ more than one copy of } v \text{ is selected} \\ (\beta/(\beta-1))k_v\mu\beta^{-\ell(v)} \;&\geq W_v^{t'} \;\geq\; c_v/(\beta+1) \;\because \text{ left bound is max possible } W_v \text{ value} \end{aligned}$$

Combining, we would have

$$\frac{c_v}{\alpha(\beta+1)} \;=\; c_v^* \;>\; k_v\mu\beta^{-\ell(v)} \;\geq\; \frac{c_v(\beta-1)}{\beta(\beta+1)},$$

so that $\alpha < \beta/(\beta-1)$. A contradiction occurs.
Thus, the lemma follows. ◀

The above lemma states that if we choose $\alpha \geq \beta/(\beta-1)$, then level down of v occurs only when $\lceil |\delta^{t_0}(v)|/k_v \rceil$ is one. Then, Case 2 inside the proof of Lemma 14 will not occur, so that we can strengthen Lemma 14 to get $|\Delta\textsc{Count}| \leq D_v^{t_0}(0,i) < \beta^i c_v^*/\mu$. Similarly, the proof of Lemma 17 can be revised, so that we can strengthen Lemma 17 by replacing h with one. On the other hand, we need $\alpha \geq (2\beta+1)/\beta + 2\epsilon$ to satisfy the amortized cost analysis. Consequently, we set $\alpha = (2\beta+1)/\beta + 2\epsilon$, and we can achieve the desired bound $\Delta B \geq \beta^i c_v^*/\mu \geq |\Delta\textsc{Count}|$. The proof for the level down case is complete.

3.3 Summary and extensions

With the appropriate setting of $\alpha = (2\beta+1)/\beta + 2\epsilon$, where $0 < \epsilon < 1$, we get an $\alpha(\beta+1)$-tight level scheme. Then, by setting $\beta = 2.43$, Theorem 8 is proven so that we get an approximation solution of ratio close to 36 with $O((\log n)/\epsilon)$ amortized update time. Note that if we focus on the non-capacitated case, that is, each vertex is weighted and has unlimited capacity,

the problem becomes the *weighted* vertex cover problem. Our dynamic scheme can easily be adapted to maintain an approximate solution, based on the following changes. First, we define the weight of a vertex W_v as $W_v = \sum_{e \sim v} \mu\beta^{-\ell(e)}$. Next, we let $\alpha = 1 + 3\epsilon$ and $\beta = 1 + \epsilon$ and revise $\phi(e)$ as $\phi(e) = (1 + \epsilon)(L - \ell(e))$. After these changes, we can go through a similar analysis, and obtain a $(2 + \epsilon)$-approximate weighted vertex cover with $O(\log n/\epsilon^2)$ amortized update time.

Finally, we consider two natural extensions of the capacitated vertex cover problem, and show how to adapt the proposed level scheme to handle these extensions.

Capacitated set cover. First, we consider the capacitated set cover problem, which is equivalent to the capacitated vertex cover problem in hyper-graphs. A hyper-graph $G = (V, E)$ has $|V| = n$ vertices and $|E| = m$ hyper-edges, where each hyper-edge is incident to a set of vertices. Suppose each hyper-edge is incident to at most f vertices. In this case, we can obtain a level scheme that maintains an $O(f^2)$ approximation solution to the dynamic set cover problem with $O(f \log(m + n)/\epsilon)$ amortized update time.

Capacitated vertex cover with non-uniform unsplittable demand. Next, we consider a more general case of the capacitated vertex cover problem in which each edge has an unsplittable demand. That is, the demand of each edge must be covered by exactly one of its endpoints. In this case, we found that it is difficult to adapt the proposed level scheme and derive similar results as before. Briefly speaking, one may re-design the weight of a vertex W_v to keep the approximation ratio, by then it becomes hard to cope with the edge insertion and deletion and maintain the $O(\log n)$ amortized update time. Nevertheless, we present two simple algorithms, one with $O(\log_2 d_{\max})$ approximation ratio and $O(\log k_{\max}/\epsilon)$ amortized update time, where $d_{\max} = \max_e\{d_e\}$, $k_{\max} = \max_v\{k_v\}$, and another with $O(1)$ approximation ratio and $O(d_{\max} \log k_{\max}/\epsilon)$ amortized update time, by re-using our proposed scheme for capacitated vertex cover.

4 Concluding Remarks

We have extended dynamic vertex cover to the more general WMCVC problem, and developed a constant-factor dynamic approximation algorithm with $O(\log n/\epsilon)$ amortized update time, where n is the number of the vertices. Note that, with minor adaptions to the greedy algorithm reported in Gupta et al.'s very recent paper [10] is also able to work for the dynamic capacitated vertex cover problem, but only to obtain a *logarithmic*-factor approximation algorithm with $O(\log n)$ amortized update time. Moreover, our proposed algorithm can also be extended to solve the soft capacitated set cover problem, and the capacitated vertex cover problem with non-uniform unsplittable edge demand.

We conclude this paper with some open problems. First, recall that in the *static* model, the soft capacitated vertex cover problem [9] can be approximated within a factor of two and three for the uniform and non-uniform edge demand cases, respectively. Here, we have shown that it is possible to design a dynamic scheme with $O(1)$ approximation ratio with polylogartihmic update time for the uniform edge demand case. Thus, designing an $O(1)$-approximation ratio algorithm with $O(\log k_{\max})$, or polylogarithmic, update time for the non-uniform edge demand case seems promising.

Recall that in the non-capacity case, both of [4, 10] achieved a large constant approximation ratio (≈ 1000) with $O(1)$ amortized update time. However, when applying their approaches directly, it seems hard to remove the coefficient h, so that the approximation ratio may be

up to $O(n)$. On the other hand, very recently, Bhattacharya et al. [3] derived a scheme with polylogartihmic *worst-case* update time and $(2+\epsilon)$-approximation ratio. They created six *states* for dynamic updates. Nevertheless, we cannot extend their approach directly, since some of these states do not satisfy the capacity constraint. It would be of significant to consider the above approaches to soft capacity vertex cover.

Another open problem is to consider *hard* capacity vertex cover, where most of the previous studies in the literature used different techniques, such as *rounding* and *patching*, from the primal-dual approach in this paper. It would be worthwhile to explore the dynamic model with hard capacity constraints.

References

1 A. Andersson and M. Thorup. Dynamic ordered sets with exponential search trees. Journal of the ACM (JACM), Vol. 54, Issue 3, No. 13 , 2007.

2 S. Baswana, M. Gupta, and S. Sen. Fully dynamic maximal matching in $O(\log n)$ update time. SIAM J. Comput. 44(2015), no. 1, pp. 88–113.

3 S. Bhattacharya, D. Chakrabarty, and M. Henzinger. Fully dynamic approximate maximum matching and minimum vertex cover in $O(\log^3 n)$ worst case update time. In Proc. the 28th ACM-SIAM Symposium on Discrete Algorithms (SODA), Barcelona, Spain, 2017, pp. 470–489.

4 S. Bhattacharya, D. Chakrabarty, and M. Henzinger. Deterministic fully dynamic approximate vertex cover and fractional matching in $O(1)$ amortized update time. In Proc. the 19th Conference on Integer Programming and Combinatorial Optimization (IPCO), Waterloo, Canada, 2017, pp. 86–98.

5 S. Bhattacharya, M. Henzinger, and G. F. Italiano. Deterministic fully dynamic data structures for vertex cover and matching. In Proc. the 26th ACM-SIAM Symposium on Discrete Algorithms (SODA), Philadelphia, USA, 2015, pp. 785–804.

6 S. Bhattacharya, M. Henzinger, and G. F. Italiano. Design of dynamic algorithms via primal-dual method. In Proc. the 42nd International Colloquium on Automata, Languages, and Programming (ICALP), Heidelberg, Germany 2015, pp. 206–218.

7 J. Chuzhoy, J. Naor. Covering problems with hard capacities. In Proc. the 43rd IEEE Symposium on Foundations of Computer Science (FOCS), 2002, pp. 481—489.

8 C. Demetrescu and G. F. Italiano. A new approach to dynamic all pairs shortest paths. Journal of the ACM (JACM), Vol. 51, Issue 6, 2004, pp. 968–992.

9 S. Guha, R. Hassin, S. Khuller, and E. Or. Capacitated vertex covering. Journal of Algorithms, Vol. 48, Issue 1, August 2003, pp. 257–270.

10 A. Gupta, R. Krishnaswamy, A. Kumar, and D. Panigrahi. Online and dynamic algorithms for set cover. In Proc. the 49th ACM Symposium on Theory of Computing (STOC), Montreal, Canada, 2017, pp. 537–550.

11 M. T. J. Holm, K. de. Lichtenberg. Poly-logarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge, and biconnectivity. Journal of the ACM (JACM) Vol. 48 Issue 4, 2001, pp. 723–760.

12 Z. Ivkovic and E. L. Lloyd. Fully dynamic maintenance of vertex cover. In Proc. the 19th International Workshop on Graph-theoretic Concepts in Computer Science (WG), London, UK, 1994, pp. 99–111.

13 O. Neiman and S. Solomon. Simple deterministic algorithms for fully dynamic maximal matching. In Proc. the 45th ACM Symposium on Theory of Computing (STOC), Palo Alto, USA, 2013, pp. 745–754.

14 K. Onak and R. Rubinfeld. Maintaining a large matching and a small vertex cover. In Proc. the 42nd ACM Symposium on Theory of Computing (STOC), Cambridge, USA, 2010, pp. 457–464.

15 D. Peleg and S. Solomon. Dynamic $(1+\epsilon)$-approximate matchings: a density-sensitive approach. In Proc. the 27th ACM-SIAM Symposium on Discrete Algorithms (SODA), Virginia, USA, 2015, pp. 712–729.

16 S. Solomon. Fully dynamic maximal matching in constant update time. In Proc. the 57th Symposium on Foundations of Computer Science (FOCS), New Jersey, USA, 2016, pp. 325–334.

Fixed-Parameter Approximation Schemes for Weighted Flowtime

Andreas Wiese[1]
Department of Industrial Engineering and Center for Mathematical Modeling,
Universidad de Chile, Chile
awiese@dii.uchile.cl

Abstract

Given a set of n jobs with integral release dates, processing times and weights, it is a natural and important scheduling problem to compute a schedule that minimizes the sum of the weighted flow times of the jobs. There are strong lower bounds for the possible approximation ratios. In the non-preemptive case, even on a single machine the best known result is a $O(\sqrt{n})$-approximation which is best possible. In the preemptive case on m identical machines there is a $O(\log \min\{\frac{n}{m}, P\})$-approximation (where P denotes the maximum job size) which is also best possible.

We study the problem in the parametrized setting where our parameter k is an upper bound on the maximum (integral) processing time and weight of a job, a standard parameter for scheduling problems. We present a $(1+\epsilon)$-approximation algorithm for the preemptive and the non-preemptive case of minimizing weighted flow time on m machines with a running time of $f(k, \epsilon, m) \cdot n^{O(1)}$, i.e., our combined parameters are k, ϵ, and m. Key to our results is to distinguish time intervals according to whether in the optimal solution the pending jobs have large or small total weight. Depending on this we employ dynamic programming, linear programming, greedy routines, or combinations of the latter to compute the schedule for each respective interval.

2012 ACM Subject Classification Theory of computation → Scheduling algorithms

Keywords and phrases Scheduling, fixed-parameter algorithms, approximation algorithms, approximation schemes

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.28

1 Introduction

A typical setting in scheduling is that one is given a set of n jobs J where each job $j \in J$ is characterized by a release date $r_j \in \mathbb{N}$, a processing time $p_j \in \mathbb{N}$, and a weight $w_j \in \mathbb{N}$. One seeks to compute a preemptive or non-preemptive schedule on m machines in which no job j is processed before its release date r_j and where each job can be processed by at most one machine at a time. Throughout this paper we will assume that the machines are identical. A natural objective function is to minimize the sum of the weighted flow times of the jobs: for C_j being the completion time of job j, the objective function is to minimize $\sum_j w_j(C_j - r_j)$.

Weighted flow time is a well-studied objective function. In the preemptive setting, the best known result on one machine is a $O(\log \log P)$-approximation algorithm [5] (where P denotes the maximum processing time of a job in the given instance) and for multiple machines in the unweighted case there is a $O(\log \min\{\frac{n}{m}, P\})$-approximation [15] and there is a lower bound of $\Omega(\log^{1-\epsilon} P)$ [11]. In the non-preemptive setting, even on one machine the best known approximation ratio is only $O(\sqrt{n})$ and this is best possible [13]. Hence, one can achieve only

[1] This work was partially supported by the Millennium Nucleus Information and Coordination in Networks ICM/FIC RC130003 and the grant Fondecyt Regular 1170223.

superconstant approximation ratios, apart from the preemptive case on one machine where in a recent break-through result a pseudo-polynomial time $O(1)$-approximation algorithm was found [6] and it is open to get a polynomial time $O(1)$-approximation (or even a PTAS) which is considered a long-standing important open problem.

Apart from approximation algorithms, another way to approach NP-hard problems is fixed-parameter tractability (FPT). One identifies a parameter k which intuitively measures the difficulty of a given instance and designs an exact algorithm with a running time of $f(k)n^{O(1)}$ for some computable function f. This is particularly appealing in settings like ours where there are strong lower bounds for the possible ratio of an approximation algorithm. Throughout this paper, we define $k := \max_j \max\{p_j, w_j\}$ which is a standard parameter in the literature for fixed-parameter algorithms for scheduling problems, see [18, 14]. Note that practical instances might have only a bounded range of job processing times and weights which makes this parameter relevant in these settings.

Recently, researchers started to combine the paradigms of approximation algorithms and fixed-parameter tractability (see, e.g., the survey by Marx [17]). Here, one constructs approximation algorithms that run in FPT time. For example, one constructs a $(1+\epsilon)$-approximation algorithm for instances with parameter k with a running time of $f(k,\epsilon) \cdot n^{O(1)}$ or $f(k) \cdot n^{O_\epsilon(1)}$, where $O_\epsilon(1)$ denotes a value that depends only on ϵ. This is in particular particularly interesting in cases where there can be no polynomial time $(1+\epsilon)$-approximation algorithm for arbitrary instances or where such an algorithm seems difficult to obtain. In this paper, we present such an algorithm for the weighted flow time problem where our combined parameter consists of the number of machines m, the quantity ϵ in the approximation ratio, and the value k as defined above.

1.1 Our contribution

We present $(1+\epsilon)$-approximation algorithms for the preemptive and the non-preemptive cases of minimizing weighted flow time on m machines with running times of $(mk)^{O(mk^3/\epsilon)} \cdot n^{O(1)}$ (preemptive case) and $(mk/\epsilon)^{O(mk^5)} \cdot n^{O(1)}$ (non-preemptive case), assuming that $p_j \leq k$ and $w_j \leq k$ for each job j. In particular, we obtain substantially better approximation factors than the known ratios of $O(\log\log P)$, $O(\log\min\{\frac{n}{m}, P\})$, and $O(\sqrt{n})$ for the respective settings.

To obtain our result in the preemptive case, the high level idea is the following: for each t denote by $W_{OPT}(t)$ the total weight of the pending jobs at time t in the optimal solution OPT, i.e., the total weight of the jobs that have been released but that have not yet been completed by time t. For time intervals during which $W_{OPT}(t)$ is small, in the optimal solution there can be only few pending jobs (bounded by a function of our parameters) and we can compute the optimal solution via a dynamic program. For the other intervals (during which $W_{OPT}(t)$ is large) we compute a preemptive schedule using a linear program. The LP is not exact since it uses a fractional objective function. However, we can guarantee that $W_{ALG}(t)$, the total weight of the pending jobs at time t in the computed solution ALG, is bounded by $W_{OPT}(t) + g(k,\epsilon,m)$ for some function g. Since $W_{OPT}(t)$ is large, the latter term is bounded by $(1+\epsilon)W_{OPT}(t)$. It is well-known that the objective function value of OPT equals $\int_t W_{OPT}(t)dt$. Therefore, we can argue that $ALG = \int_t W_{ALG}(t)dt \leq \int_t (1+\epsilon)W_{OPT}(t)dt = (1+\epsilon)OPT$. Since we do not know beforehand the values for t where $W_{OPT}(t)$ is large and small, we devise a dynamic program that scans the time axis from left to right step by step, guesses the schedules for the time intervals during which $W_{OPT}(t)$ is small, and computes the LP-solution for all intervals during which $W_{OPT}(t)$ is large. This yields a $(1+\epsilon)$-approximation for the preemptive setting.

In the non-preemptive case we again distinguish time intervals depending on whether $W_{OPT}(t)$ is large or small. For values of t where $W_{OPT}(t)$ is small, like before there are only few options for the pending jobs and, intuitively, we use a dynamic program to compute the optimal solution. For intervals I during which $W_{OPT}(t)$ is large we can no longer use the LP since it produces a schedule that is possibly preemptive. Even more, given I and a set of jobs J' that we want to complete within I (some of them might be released during I), it is not even clear how to determine whether there exists is a non-preemptive schedule for J' in I, even independent of the cost of such a schedule! To this end, we introduce the following strategy. If at a time t many jobs of the same *type* are pending, i.e., jobs with the same processing time and weight, then we schedule a large subset of them as soon as possible after time t on each machine. We say that they are scheduled as a *pack*. In case that there are several job types with many pending jobs, we give priority to the job type with the highest Smith ratio, i.e., with the largest ratio of weight divided by processing time. We show by an exchange argument that there indeed exists a feasible non-preemptive schedule that follows this rule at each time t (though we do not bound its cost at this point).

To bound our cost during an interval in which $W_{OPT}(t)$ is large, for the jobs scheduled as packs, we compare their cost with an optimal fractional schedule for them and prove that at each time t the total weight of the pending jobs in the two schedules differs by at most an additional term $\bar{g}(k, \epsilon, m)$ (for some function $\bar{g}$) which we can bound by $\epsilon W_{OPT}(t)$. For all other scheduled jobs we can show that each time t only few of them are pending and thus we can bound their total weight by another term $\epsilon W_{OPT}(t)$. This implies that $W_{ALG}(t) \leq (1+O(\epsilon))W_{OPT}(t)$ for each time t and hence our solution is $(1+\epsilon)$-approximative. We again devise a DP that scans the time axis from left to right and computes the solution ALG step by step, yielding a $(1+\epsilon)$-approximation algorithm for the non-preemptive case.

Note that in the non-preemptive case, already on one machine and with identical job weights the problem is NP-hard [13]. Hence, it is necessary to require that $p_j \leq k$ for each job j for the problem to be FPT in this case. On the other hand, no $W[1]$-hardness results are known for the settings of our FPT-algorithms above. Due to space constraints most proofs had to be omitted or moved to the appendix.

1.2 Other related work

For a single machine in the preemptive setting there is a QPTAS if the processing times and weights are in a quasi-polynomial range [8]. Its running time is $n^{O(\log W \log P/\epsilon^3)}$ (where W denotes the maximum job weight in the instance) and it is based on grouping the jobs according to processing times and weights. Note that the number of different groups appears in the exponent of n and hence we cannot use its methodology to obtain a running time of the form $f(k,\epsilon)n^{O(1)}$ or $f(k)n^{O_\epsilon(1)}$. The latter result was generalized by Bansal to an algorithm for a constant number of unrelated machines in the case when preemption is allowed by migration is forbidden, having a running time of $2^{(m \min\{\log nW, \log nP\}^3/\epsilon^3)}$ [3]. If $P \leq k$ and $W \leq k$ (like in our parametrized setting) this yields a running time of $2^{(m(\log nk)^3/\epsilon^3)}$ while our algorithms above for the preemptive case *with* migration and for the non-preemptive case (both being incomparable with Bansal's setting) have running times of $(mk)^{O(mk^3/\epsilon)} \cdot n^{O(1)} = 2^{O(\log(mk)\cdot mk^3/\epsilon + O(\log n))}$ and $(mk/\epsilon)^{O(mk^5)} \cdot n^{O(1)} = 2^{O(\log(mk/\epsilon)\cdot mk^5 + O(\log n))}$, respectively.

In the non-preemptive setting, on m machines one can achieve a ratio of $O(\sqrt{n/m}\log(n/m))$ and there is a lower bound of $\Omega(n^{1/3-\epsilon})$ [15]. Minimizing flow time was also studied in the online setting. In the unweighted case on one machine the SRPT algorithm is optimal and on parallel machines it achieves a competitive ratio of $O(\log \min\{\frac{n}{m}, P\})$ which is best possible [15, 11]. In the weighted case, for one machine there is an $O(\log^2 P)$-

competitive semi-online algorithm known [9] and for multiple machines there is a lower bound of $\Omega(\min\{\sqrt{P}, \sqrt{W}, (n/m)^{1/4}\})$ [9]. This was improved recently to a $O(\log(\min\{P, D, W\}))$-competitive algorithm [2] (where D is the maximum ratio of the job densities), using the $O(\log D)$-competitive algorithm from [4]. Many results are known in the resource augmentation setting in which the online algorithm is given faster machines than the adversary, see [16, Chapter 15] and references therein.

If all jobs are released at time zero, the problem is equivalent to minimizing the weighted sum of completion times. This problem admits a polynomial time $(1+\epsilon)$-approximation algorithm and in the preemptive case even an EPTAS [1]. This was generalized to PTASs for related machines [7] for which in the non-preemptive case there is also an EPTAS [10].

Approximation schemes with a running time of $f(k) \cdot n^{O_\epsilon(1)}$ (where the parameter k denotes the size of the desired solution) are known for Maximum Independent Set of Rectangles, Two-dimensional Geometric Knapsack with rotations [12], and for Unsplittable Flow on a Path [19] while PTASs are open for these problems.

2 Preemptive case

We present our $(1+\epsilon)$-approximation algorithm for minimizing the weighted flow time on m identical machines with preemption, i.e., $P|r_j, pmtn|\sum w_j F_j$. Our algorithm has a running time of $k^{O(m \cdot k^3/\epsilon)} n^{O(1)}$, assuming that $p_j, w_j \in \{1, ..., k\}$ for each job j, i.e., our combined parameter is $k + m + 1/\epsilon$. We proceed in two steps: first, we show that there is a structured solution OPT' such that $c(OPT') \leq (1+\epsilon)c(OPT)$. Then, we devise a dynamic program that computes a solution with this structure whose cost is at most the cost of OPT'.

We group the jobs into groups. For each $\ell, \ell' \in \{1, ..., k\}$ we define $J_{\ell,\ell'} := \{j | p_j = \ell \wedge w_j = \ell'\}$. First, we prove by some simple exchange arguments that for each job group the optimal solution OPT essentially processes its jobs in the order of their release dates and hence the number of partially processed jobs of each group is small. Throughout this paper, whenever we speak about the order of the release dates of the jobs, we break ties in a fixed arbitrary order. Also, we will assume that $1/\epsilon \in \mathbb{N}$ and define $\Gamma := \cup_{t \in \mathbb{N}_0} \epsilon t$.

▶ **Lemma 1.** *By losing a factor of $1+\epsilon$ in the approximation ratio, we can assume the following properties for OPT: for each $t \in \Gamma$, each machine i works on exactly one job during the entire interval $[t, t+\epsilon)$ or is idle during $[t, t+\epsilon)$. Also, for each job class $J_{\ell,\ell'}$ and each time t, the jobs of $J_{\ell,\ell'}$ that are partially but not completely scheduled at time t are among the mk/ϵ jobs with the earliest release dates pending at time t.*

For ease of notation, in the sequel we will denote by OPT the $(1+\epsilon)$-approximative solution due to Lemma 1.

2.1 Near-optimal solution

We define a near-optimal structured solution OPT'. For a schedule S and any $t \geq 0$ let $W_S(t)$ denote the total weight of the jobs j that satisfy $r_j \leq t$ but that are not finished by time t in S. We define $I_{\text{low}} := \bigcup_{t \in \Gamma : W_{OPT}(t) \leq 3mk^3/\epsilon^2} [t, t+\epsilon)$ and $I_{\text{high}} := \bigcup_{t \in \Gamma : W_{OPT}(t) > 3mk^3/\epsilon^2} [t, t+\epsilon)$. For each time interval $[t, t+\epsilon) \subseteq I_{\text{low}}$ with $t \in \Gamma$ we define the schedule OPT' to be identical to OPT. For the timesteps in I_{high} we consider each maximally large interval in I_{high}. Let $[t_1, t_2) \subseteq I_{\text{high}}$ be such an interval with $t_1, t_2 \in \Gamma$. In order to define OPT' during $[t_1, t_2)$, we compute a schedule using a linear program. For each job j, denote by $p'_j \in \mathbb{N}$ the amount of work of j that OPT finishes during $[t_1, t_2)$. Note that p'_j is an integral multiple of ϵ. Our linear program computes a schedule for $[t_1, t_2)$ such that we finish exactly p'_j units of work

of each job j during $[t_1, t_2)$, at each time $t \in [t_1, t_2)$ each job j is processed by at most one machine, and the objective function is a fractional objective in which we assume that we split each job j into p'_j/ϵ unit size pieces of weight $\epsilon w_j/p'_j$ each. This yields the following linear program that we denote by (LP).

$$\begin{aligned}
\min \quad & \sum_{j \in J} \sum_{t \in \{t_1, t_1+\epsilon, \dots, t_2-\epsilon\}} x_{ijt} \cdot (t+\epsilon)\epsilon \frac{w_j}{p'_j} \\
\text{s.t.} \quad & \sum_{i=1}^{m} \sum_{t \in \{t_1, t_1+\epsilon \dots, t_2-\epsilon\}} x_{ijt} = p'_j \quad \forall j \in J \\
& \sum_{i=1}^{m} x_{ijt} \leq 1 \quad \forall j \in J \, \forall t \in \{t_1, t_1+\epsilon, \dots, t_2-\epsilon\} \\
& \sum_{j \in J} x_{ijt} \leq 1 \quad \forall i \in [m] \, \forall t \in \{t_1, t_1+\epsilon, \dots, t_2-\epsilon\} \\
& x_{ijt} \geq 0 \quad \forall j \in J \, \forall i \in [m] \, \forall t \in \{r_j, r_j+\epsilon, \dots, t_2-\epsilon\}
\end{aligned}$$

We compute an optimal extreme point solution to (LP) in polynomial time. By interpreting (LP) as a model for a suitable instance of bipartite matching one can easily argue that this solution is integral. We simplify it to ensure that at each time at most mk/ϵ jobs from the same group are partially processed (where "partially" is defined with respect to the processing time p'_j), using some exchange arguments.

▶ **Lemma 2.** *Given an optimal integral solution to (LP), in time polynomial in n and k we can compute an optimal integral solution that has the property that at each time $t \in \{t_1, t_1+\epsilon, \dots, t_2-\epsilon\}$ from each job class $J_{\ell,\ell'}$ there are at most mk/ϵ jobs j such that $0 < \bar{p}_j(t) < p'_j$ where $\bar{p}_j(t)$ denotes the amount of j that has been processed during $[t_1, t)$.*

The schedule of OPT' during $[t_1, t_2)$ is now defined by applying Lemma 2 to the optimal integral solution to (LP). For x being the resulting solution, during an interval $[t, t+\epsilon) \subseteq [t_1, t_2)$ with $t \in \Gamma$ we schedule a job j on a machine i if and only if $x_{ijt} = 1$. We do the above procedure for each maximally large interval $[t_1, t_2) \subseteq I_{\text{high}}$ with $t_1, t_2 \in \Gamma$. This completes the definition of OPT'.

In order to show that OPT' has small cost compared to OPT, we use the following well-known way to measure the flow time objective function. We define the cost of a schedule S to be $c(S) := \sum_{j \in J} w_j(C_j^S - r_j)$ where C_j^S denotes the completion time of a job j in S.

▶ **Proposition 3.** *For any schedule S we have that $c(S) = \int_t W_S(t)dt$.*

By construction have that $W_{OPT'}(t) = W_{OPT}(t)$ for each $t \in I_{\text{low}}$. In each maximally large interval $[t_1, t_2) \subseteq I_{\text{high}}$ the solution OPT' equals the solution to (LP) which is the optimal solution for the fractional objective function. The latter differs from the integral objective function since there can be jobs that are partially but not completely processed. However, due to Lemmas 1 and 2 there can be at most $3mk/\epsilon$ such jobs from each class $J_{\ell,\ell'}$ and thus their total weight is bounded by $3mk^3/\epsilon$. However, $W_{OPT}(t) > 3mk^3/\epsilon^2$ if $t \in I_{\text{high}}$ and hence the total contribution of these jobs to $W_{OPT'}(t)$ is bounded by $\epsilon W_{OPT}(t)$ for each $t \in \Gamma$. Using Proposition 3 and additionally that the fractional cost of the LP-schedule is at most the cost of OPT in the same respective interval $[t_1, t_2)$, we can prove the following lemma.

▶ **Lemma 4.** *It holds that $c(OPT') \leq (1+\epsilon)c(OPT)$.*

2.2 Dynamic program

We give a dynamic program that computes a schedule whose cost is at most $c(OPT')$. It scans the time axis from left to right. Given a timestep $t \in \Gamma$ with $t \in I_{\text{low}}$ it intuitively guesses the schedule of OPT during $[t, t+\epsilon)$. If $t+\epsilon \in I_{\text{low}}$ then it proceeds with the timestep $t+\epsilon$. If $t+\epsilon \in I_{\text{high}}$ then it guesses the earliest timestep $t' \in \Gamma$ such that $t < t'$ and $t' \in I_{\text{low}}$, and additionally it guesses the characteristics of the pending jobs at time t'. For the time interval $[t, t')$ it computes a schedule using (LP). At each time $t \in I_{\text{low}}$ there are at most $3mk^3/\epsilon^2$ pending jobs in OPT and due to Lemma 1 there are only $f(k, m, \epsilon)$ many possibilities for their characteristics, for some function f. Hence, we can embed the whole procedure into a dynamic program that has $f(k, \epsilon, m)$ cells for each time $t \in \Gamma$.

We assume w.l.o.g. that $\min_j r_j = 1$ and that $\max_j r_j = O(n^2 k)$ since we can shift the release dates uniformly and if $\max_j r_j$ is too large then we can split the instance into independent subinstances. Note that then OPT' finishes its last job by time $T := \max_j r_j + nk$ the latest.

We define now our dynamic program formally. We say that a *configuration for a time* $t \in \Gamma$ specifies for each job its remaining processing time at time t. Formally, a configuration is a vector $(\bar{p}_j)_{j \in J}$ such that $0 \leq \bar{p}_j \leq p_j$ and $\bar{p}_j/\epsilon \in \mathbb{N}$ for each $j \in J$. For each time $t \in \Gamma$ there is a configuration that corresponds to the pending jobs of OPT at time t. We will show now that if $t \in I_{\text{low}}$ then there are only $mk^{O(mk^3/\epsilon)}$ possibilities for this configuration, using that there can be only $3mk^3/\epsilon^2$ pending jobs at time t.

▶ **Lemma 5.** *For each time $t \in \Gamma$ we can compute in time $(mk)^{O(mk^3/\epsilon)} n^{O(1)}$ a set $\mathcal{C}(t)$ of at most $(mk)^{O(mk^3/\epsilon)}$ configurations such that if $t \in I_{\text{low}} \cap \Gamma$ then one of them corresponds to OPT at time t. The set $\mathcal{C}(T)$ contains only the configuration c_T in which all jobs have completed.*

Proof. If $t \in I_{\text{low}} \cap \Gamma$ then $W_{OPT}(t) \leq 3mk^3/\epsilon^2$ and hence there can be at most $3mk^3/\epsilon^2$ jobs pending at time t. For each of them there are only k^2 options which class it belongs to, k/ϵ possibilities for how much processing time is left, and then by Lemma 1 only $3mk^3/\epsilon^2 + mk/\epsilon$ possibilities for its identiy. This yields $(k^3/\epsilon(3mk^3/\epsilon^2 + mk/\epsilon))^{3mk^3/\epsilon^2} = (mk)^{O(mk^3/\epsilon)}$ combinations. For each of them we can compute the corresponding set of jobs in time $n^{O(1)}$, yielding a running time of $(mk)^{O(mk^3/\epsilon)} n^{O(1)}$ for the computation of all combinations. ◀

Our dynamic program has a cell (t, c) for each combination of a time $t \in \Gamma \cap [0, ..., T]$ and a configuration $c \in \mathcal{C}(t)$. We say that a job is *pending in* c if it has not yet been completed according to c. The entry of the cell (t, c) stores a schedule for the time interval $[t, T)$ in which the remaining parts of the pending jobs in c are scheduled and additionally all jobs j with $r_j > t$. Then the cell $(0, c_0)$ stores a schedule for the whole instance where c_0 denotes the configuration in which no job has been worked on yet, i.e., $\bar{p}_j = p_j$ for each job j.

For the base case, we assign to the cell (T, c_T) the empty schedule. Suppose we are given a cell (t, c) with $t < T$. We compute its schedule as follows. The reader may imagine that $t \in I_{\text{low}}$ and that c is the configuration of OPT' (and hence OPT) at time t. Intuitively, we guess the schedule of OPT' during $[t, t+\epsilon)$. Formally, we try all possibilities for up to m jobs $j_1, ..., j_{m'}$ that are pending in c and that are among the mk/ϵ jobs with earliest release dates among the pending jobs at time t of their respective job class (note that we can assume that OPT' does not schedule other jobs during $[t, t+\epsilon)$, see Lemma 1). In particular, the number of schedules to enumerate is bounded by $g(k, \epsilon, m)$ for some function g. Each guess of a set of jobs $j_1, ..., j_{m'}$ to be processed during $[t, t+\epsilon)$ yields a configuration c' for time $t+\epsilon$. If $c' \in \mathcal{C}(t+\epsilon)$ then we append the schedule stored in $DP(t+\epsilon, c')$ to the schedule

for $[t, t+\epsilon)$. Note that in case that OPT' is in configuration c at time t and we guessed $j_1, ..., j_{m'}$ correctly, for the interval $[t, t+\epsilon)$ we obtain the same schedule as OPT' (up to permutation of machines) and at time $t+\epsilon$ the schedule OPT' is in configuration c'.

If $c' \notin \mathcal{C}(t+\epsilon)$ then we guess the smallest time $t'' \in \Gamma$ with $t'' > t$ such that $t'' \in I_{\text{low}}$ and additionally the configuration c'' of OPT' at time t''. For the time interval $[t+\epsilon, t'')$ we compute a schedule using (LP): for each job $j \in J$ denote by $\bar{p}'_j$ its remaining processing time at time $t+\epsilon$ according to c' and by $\bar{p}''_j$ its remaining processing time at time t'' according to c''. We solve (LP) where for each job j we define $p'_j := \bar{p}'_j - \bar{p}''_j$. If the LP does not have a solution then we reject our guess. Otherwise, we apply Lemma 2 to the resulting solution. We append the schedule in $DP(t'', c'')$ to the computed schedules for $[t, t+\epsilon)$ and for $[t+\epsilon, t'')$. Note that if all guesses were correct, then for $[t, t'')$ we obtain the same schedule as OPT' (up to permutation of machines). Each enumerated guess yields a schedule for the jobs pending at time t in c (completing their remaining processing time according to c) and the jobs j with $r_j > t$ (processing them completely). Among all these solutions, we store in $DP(t, c)$ the solution that minimizes the sum of the weighted flow times of all the latter jobs.

One can show by induction over the DP-cells that the cost of the solution in the cell $(0, c_0)$ (note that $c_0 \in \mathcal{C}(0)$ since $\min_j r_j = 1$) is bounded by $c(OPT')$: whenever we process a DP-cell (t, c) such that OPT' is in configuration c at time t, among our enumerated guesses are the jobs that OPT processes during $[t, t+\epsilon)$, and t'' and c'' according to OPT' which then yields the same schedule for $[t, t'')$ as OPT'. The number of DP-cells is bounded by the total number of configurations for all relevant time steps, see Lemma 5. Moreover, we can compute the entry for each DP-cell by performing $(mk)^{O(mk^3/\epsilon)} \cdot n^{O(1)}$ guesses (for the jobs jobs $j_1, ..., j_{m'}$, t'' and c'') and a computation of $n^{O(1)}$ for each guess. This yields the following theorem.

▶ **Theorem 6.** *For the problem of minimizing the weighted flow time on m identical machines with preemption and job release dates, $P|r_j, pmtn|\sum w_j F_j$, there is a $(1+\epsilon)$-approximation algorithm with a running time of $(mk)^{O(mk^3/\epsilon)} \cdot n^{O(1)}$, assuming that $p_j, w_j \in \{1, ..., k\}$ for each job j.*

3 Non-preemptive case

In this section we present a $(1+\epsilon)$-approximation algorithm for the non-preemptive setting, i.e, for $P|r_j|\sum w_j F_j$, whose running time is bounded by $f(k, \epsilon, m) \cdot n^{O(1)}$ for some function f, assuming that $p_j, w_j \in \{1, ..., k\}$ for each job j.

Similar to Lemma 1 in the preemptive case, we can assume that OPT schedules the jobs in order of their release dates.

▶ **Lemma 7.** *We can assume that if OPT starts a job $j \in J_{\ell,\ell'}$ before a job $j' \in J_{\ell,\ell'}$ then r_j is released before $r_{j'}$, breaking ties in an arbitrary fixed order. Also, each job starts and ends at an integral time point.*

3.1 Near-optimal solution

Like in the preemptive case, we first define a structured solution OPT' with $c(OPT') \leq (1+\epsilon)c(OPT)$ and then provide a dynamic program that finds a solution with cost at most $c(OPT')$. To define OPT', we split the time horizon into two parts I'_{low} and I'_{high}. We define $I'_{\text{low}} := \bigcup_{t \in \mathbb{N}: W_{OPT}(t) \leq 2^{2k^2} m^2/\epsilon} [t, t+1)$ and $I'_{\text{high}} := \bigcup_{t \in \mathbb{N}: W_{OPT}(t) > 2^{2k^2} m^2/\epsilon} [t, t+1)$. For each time step $[t, t+1) \subseteq I'_{\text{low}}$ with $t \in \mathbb{N}$ we define OPT' to be identical to OPT. Let $[t_1, t_2) \subseteq I'_{\text{high}}$ be a maximally large interval of I'_{high} (and thus $t_1, t_2 \in \mathbb{N}$). Our goal is now

to define a structured schedule for $[t_1, t_2)$ whose cost is by at most a factor $1 + \epsilon$ larger than the cost of OPT during $[t_1, t_2)$. Let J' denote the jobs that in OPT are started and completed during $[t_1, t_2)$. In the preemptive case we defined a schedule for J' during $[t_1, t_2)$ via (LP). However, an LP-solution might preempt (and migrate) jobs. Instead, we look for a structured non-preemptive solution that our dynamic program can easily compute later. Note that if we are given the interval $[t_1, t_2)$ with the set of jobs J' it is not even clear how compute *any* non-preemptive schedule, even independent of the cost.

We define a structured schedule that has the *pack-property*. Intuitively, this property implies that if at some time t there are many pending jobs of the same class then we schedule a *pack*, i.e., a large subset of them, as soon as we can. Formally, we say that a job j is *ready* at time t if $r_j \leq t$ and if j has not been started before t. A set of ready jobs $Q \subseteq J'$ forms a *pack* if all jobs in Q are of the same class and if $p(Q) = m \cdot k!$, where for any set of jobs $\tilde{J}$ we define $p(\tilde{J}) := \sum_{j \in \tilde{J}} p_j$.

▶ **Definition 8** (Pack-property)**.** A schedule for the jobs J' in the interval $[t_1, t_2)$ has the *pack-property* if for each integral time $t \in [t_1, t_2)$ at which a machine finishes a job or such that there is machine that is idle during $[t-1, t)$ the following holds:

- If for some job class $J_{\ell,\ell'}$ there are jobs from $J' \cap J_{\ell,\ell'}$ with a total processing time of at least $4m^2kk!$ that are ready at time t, we say that the class $J_{\ell,\ell'}$ is *waiting*. For a waiting job class $J_{\ell,\ell'}$ let $Q_{\ell,\ell'}$ denote the $m \cdot k!/\ell$ ready jobs with earliest release dates among the ready jobs in $J_{\ell,\ell'}$.
- Among the waiting jobs classes, let $J_{\ell,\ell'}$ be a waiting job class with highest density, i.e., that maximizes ℓ'/ℓ, breaking ties in a fixed arbitrary way. For each machine i let $t(i)$ denote the earliest time $t' \geq t$ when i is not processing a job that it is executing during $[t-1, t)$. We require that then each machine i schedules $k!/\ell$ jobs from $Q_{\ell,\ell'}$ during $[t(i), t(i) + k!)$. In the resulting schedule, we call the jobs in $Q_{\ell,\ell'}$ *pack-jobs* and we say that they are *scheduled as the pack* $Q_{\ell,\ell'}$.

We first prove that a schedule for $[t_1, t_2)$ with the pack-property always exists and afterwards we will bound its cost. We will do this via a repeated exchange argument: starting with the optimal schedule, if at some time t the schedule does not obey the pack-property then we move some jobs in the schedule such that the pack-property holds at time t. Note that in OPT a machine might work on a job $j \notin J'$ during $[t_1, t_2)$ if j is started before t_1 or completed after t_2. For each machine i, denote by $t_1(i)$ and $t_2(i)$ the earliest and latest times in $[t_1, t_2]$ at which i does not work on a job in J'.

▶ **Lemma 9.** *There is a schedule for the jobs J' during the interval $[t_1, t_2)$ that fulfills the pack-property such that each machine i works on jobs in J' only during $[t_1(i), t_2(i))$.*

We apply Lemma 9 to each maximally large interval $[t_1, t_2) \subseteq I'_{\text{high}}$ and its corresponding job set J'. Then we possibly swap jobs from the same class such that the resulting schedule satisfies Lemma 7. Denote by OPT' the resulting schedule.

3.2 Bounding the cost

We want to prove that $c(OPT') \leq (1 + \epsilon)c(OPT)$. The intuition is the following: recall that in a schedule with the pack-property some job are pack-jobs. The *non-pack-jobs* are all other jobs that are scheduled during a maximally large interval $[t_1, t_2) \subseteq I'_{\text{high}}$, denoted by J_{np}. At each time $t \in I'_{\text{high}}$ there can be only few pending jobs in J_{np} since if the total processing time of pending non-pack jobs from one class was at least $4m^2kk!$ then some of them would be pack-jobs. Hence, the contribution to $W_{OPT'}(t)$ of the non-pack-jobs is negligible for each $t \in I'_{\text{high}}$ and hence the same holds for their contribution to the objective function.

▶ **Proposition 10.** *For each time $t \in I'_{\text{high}}$ let $W^{\text{np}}_{OPT'}(t)$ denote the total weight of the pending jobs in J_{np} at time t in OPT' and then it holds that $W^{\text{np}}_{OPT'}(t) \leq 4m^2k^3k! \leq \epsilon \cdot W_{OPT}(t)$. Therefore, $\int_{t \in I'_{\text{high}}} W^{\text{np}}_{OPT'}(t)dt \leq \epsilon \cdot OPT$.*

Let J_{p} denote the jobs that are scheduled as pack-jobs during a maximally large interval $[t_1, t_2) \subseteq I'_{\text{high}}$. When in OPT' a set of jobs $\bar{Q}_{\ell,\ell'}$ is scheduled as a pack then for each machine i we can identify a time $t(i)$ such that the jobs from $\bar{Q}_{\ell,\ell'}$ are scheduled exactly during $[t(i), t(i) + k!)$. It can happen that $t(i) \neq t(i')$ for two machines i, i'. In order to simplify the analysis, we define a new schedule OPT''_{p} only for the pack-jobs in which the latter does *not* happen and such that the cost of OPT''_{p} is similar to the cost of the pack jobs in OPT'. Intuitively, OPT''_{p} is a greedy schedule: one can interpret the jobs from each pack as m super-jobs with processing time $k!$ each such that each machine is assigned one of these super-jobs. In our schedule, these super-jobs are scheduled greedily by their density (or, equivalently, by their weight).

Let $Q_1, ..., Q_s$ be a partition of the pack-jobs such that for each r the jobs in Q_r are scheduled as the pack Q_r. For each pack Q_r denote by t_r the earliest start time of a job in Q_r in OPT'. In OPT''_{p} we define that on each machine i for each pack Q_r the jobs in Q_r are scheduled during $[t_r, t_r + k!)$. Intuitively, to achieve this we shift some of the jobs of Q_r in OPT' to the left. Denote by OPT''_{p} the resulting schedule. At time t_r all jobs from Q_r are already released and thus in OPT''_{p} no job is scheduled before its release date. Together with the following lemma this implies that OPT''_{p} is feasible.

▶ **Lemma 11.** *For any two packs $Q_r, Q_{r'}$ with $r \neq r'$ we have that $[t_r, t_r + k!) \cap [t_{r'}, t_{r'} + k!) = \emptyset$.*

Proof. Assume w.l.o.g. that $t_r \leq t_{r'}$. In OPT', for the pack Q_r there is a machine i such that the jobs from Q_r are scheduled during $[t_r, t_r + k!)$ and for any machine $i' \neq i$ the jobs from Q_r are scheduled during $[t_r + \alpha_{i'}, t_r + \alpha_{i'} + k!)$ for some $\alpha_{i'} \in \{0, ..., k-1\}$. Hence, $t_{r'} \geq t_r + k!$. ◀

When we constructed OPT''_{p} based on OPT', we shifted each job by at most $k - 1$ units to the left and hence the cost of the schedule changes only marginally.

▶ **Lemma 12.** *We have that $W^{\text{p}}_{OPT'}(t) \leq W_{OPT''_{\text{p}}}(t) + k^2m \leq W_{OPT''_{\text{p}}}(t) + \epsilon W_{OPT}(t)$ where $W^{\text{p}}_{OPT'}(t)$ denotes the total weight of the pending pack-jobs in OPT' at time t.*

Proof. Observe that in OPT' and OPT''_{p} the completion time of a job differs by at most k. Hence, at each point in time OPT''_{p} has completed at most km jobs more than OPT' and the total weight of these jobs is at most k^2m. Therefore, $W^{\text{p}}_{OPT'}(t) \leq W_{OPT''_{\text{p}}}(t) + k^2m \leq W_{OPT''_{\text{p}}}(t) + \epsilon W_{OPT}(t)$ for each $t \in I'_{\text{high}}$. ◀

We want to prove now that OPT''_{p} has small cost. Consider a maximally large interval $[t_1, t_2) \subseteq I'_{\text{high}}$. We define an artificial instance I_{p} consisting of only the pack-jobs scheduled during $[t_1, t_2)$. We assign an artificial release date t'_r to all jobs in each pack Q_r. Intuitively, t'_r is defined to be the earliest time when we might schedule the jobs in Q_r as pack-jobs because at this time all jobs in Q_r and many other jobs of the same class with later release dates have already been released. Formally, assume that $J_{\ell,\ell'}$ is the (unique) job class such that Q_r contains jobs from $J_{\ell,\ell'}$. We define t'_r to be the earliest time when all jobs in Q_r are released and the total processing time of already released jobs $j \in J_{\ell,\ell'}$ with $r_j \geq \min_{j' \in Q_r} r_j$ is at least $4m^2kk!$. Note that OPT' does not process a job in Q_r earlier than t'_r since in OPT' the jobs in Q_r are processed as pack jobs. We consider the solution to (LP) for this instance where we define $p'_j := p_j$ for each $j \in J_{\text{p}}$. Let x^* be this solution and let $FRAC$

be the resulting fractional schedule. We partition the jobs J_p into groups $J_\mathrm{p}^{(1)}, J_\mathrm{p}^{(2)}, ..., J_\mathrm{p}^{(\gamma)}$ according to their densities: for each group $J_\mathrm{p}^{(g)}$ all jobs $j \in J_\mathrm{p}^{(g)}$ have exactly the same density w_j/p_j and for two jobs j, j' with $j \in J_\mathrm{p}^{(g)}$ and $j' \in J_\mathrm{p}^{(g')}$ with $g < g'$ we have that $w_j/p_j < w_{j'}/p_{j'}$. In particular, jobs in different groups have different densities.

▶ **Lemma 13.** *We can assume w.l.o.g. that in $FRAC$ for each $t \in I'_\mathrm{high}$ with $t \in \mathbb{N}$ we have that during $[t, t+1)$ either all machines are idle or all machines work on jobs from a pack Q_r such that each job $j \in Q_r$ has the highest density w_j/p_j among all pending jobs at time t. Moreover, for each group $J_\mathrm{p}^{(g)}$ at each time t there is at most one pack Q_r containing jobs from $J_\mathrm{p}^{(g)}$ from which some jobs or some parts of some jobs have already been processed, but not all jobs completely.*

Proof. The number of jobs in each pack is $mk!/\ell$ where ℓ denotes the size of each job in the pack and, in particular, this value is divisible by m. Also, a solution to (LP) is optimal if for each $t \in I'_\mathrm{high}$ during $[t, t+1)$ each machine works on a job j with the largest density w_j/p_j among all available jobs. Hence, using a greedy algorithm we can construct an optimal fractional solution with the latter property that satisfies the claim of the lemma. ◀

For each $t \in I'_\mathrm{high}$ denote by $\tilde{W}_{FRAC}(t)$ the total fractional pending weight of the jobs j with $r_j \leq t$ that have not completed by time t in $FRAC$. Formally, for each job $j \in J_\mathrm{p}$ and each $t \in I'_\mathrm{high}$ let $p_j(t)$ denote the remaining processing time of job j at time t in $FRAC$. We define $\tilde{W}_{FRAC}(t) := \sum_{j \in J_\mathrm{p}} p_j(t) \frac{w_j}{p_j}$.

In the next lemma, we prove that $\tilde{W}_{FRAC}(t)$ is not too large compared to $W_{OPT}(t)$ for each t. Note that even though $\tilde{W}_{FRAC}(t)$ denotes the fractional remaining weight which is not larger than the integral remaining weight, for some t it can be that $W_{OPT}(t) < \tilde{W}_{FRAC}(t)$ because OPT can work on a job j in a pack Q_r before $FRAC$ can do this since possibly $r_j < t'_r$. However, for each job class $J_{\ell,\ell'}$ the total weight of such jobs is bounded by $4m^2k^3k!$ at each time t which we will use to prove the following lemma.

▶ **Lemma 14.** *For each group $J_\mathrm{p}^{(g)}$ and each time $t \in \{t_1, ..., t_2 - 1\}$ we have that $\tilde{W}_{FRAC}(t) \leq W_{OPT}(t) + 4m^2k^5k!$.*

In the next lemma, we show that $W_{OPT'_p}(t)$ is not much larger than $\tilde{W}_{FRAC}(t)$ which we will eventually use in order to bound the cost of OPT'_p and hence of OPT'. For a set of jobs $\bar{J}$ denote by $W^{\bar{J}}_{OPT''_\mathrm{p}}(t)$ and $\tilde{W}^{\bar{J}}_{FRAC}(t)$ the integral and fractional weight of its pending jobs at time t in OPT'_p and $FRAC$, respectively.

▶ **Lemma 15.** *For each group $J_\mathrm{p}^{(g)}$ and each $t \in \{t_1, ..., t_2 - 1\}$ we have that $W^{J_\mathrm{p}^{(g)}}_{OPT''_\mathrm{p}}(t) \leq \tilde{W}^{J_\mathrm{p}^{(g)}}_{FRAC}(t) + 2^{\gamma-g}mk \cdot k!$. Therefore, $W_{OPT''_\mathrm{p}}(t) \leq \tilde{W}_{FRAC}(t) + 2^{k^2}mk \cdot k! \leq W_{OPT''_\mathrm{p}}(t) + \epsilon W_{OPT}(t)$.*

Using Lemmas 14 and 15 we bound the cost of OPT'.

▶ **Lemma 16.** *It holds that $c(OPT') \leq (1+\epsilon)c(OPT)$.*

3.3 Dynamic program

We present now a dynamic program that computes a solution ALG which satisfies that $c(ALG) \leq c(OPT') \leq (1+\epsilon)c(OPT)$. It has two stages: in the first stage, it intuitively

guesses the schedule for each $t \in I'_{\text{low}}$ and the maximally large intervals of I'_{high}. Note that for each $t \in I'_{\text{low}}$ we have that $W_{OPT}(t) \leq 2^{2k^2}m^2/\epsilon$ and hence the number of different configurations at time t can be bounded by some function $f(k, \epsilon, m)$. In the second stage, it computes a solution for each maximally large subinterval of I'_{high} that obeys the pack-property. To this end, note that if at a time τ during such a subinterval $[\tau_L, \tau_R)$ no job class is waiting then the number of released by infinished jobs at time τ can be bounded by some function $g(k, \epsilon, m)$. Thus, we can guess the schedule for the interval $[\tau, \tau + 1)$ in FPT-time and continue with the time point $\tau + 1$. Otherwise, the pack-property dictates which jobs we need to schedule next on the machines, i.e., we need to schedule a pack of jobs from the waiting job class of highest density (among all waiting job classes). We embed the guessing steps into a dynamic program that intuitively scans the time axis from left to right.

We assume w.l.o.g. that $\min_j r_j = 1$, $\max_j r_j = O(n^2k^2)$, and that OPT' finishes its last job before time $T := \max_j r_j + nk$, otherwise we can split the instance into independent subinstances. Similarly to the preemptive setting, we say that a *configuration for a time t* specifies which jobs are pending at time t, which jobs are being processed on some machine at time t, and the remaining processing time for each job of the latter kind at time t. Formally, a configuration is a vector $(\bar{p}_j)_{j \in J}$ such that $0 \leq \bar{p}_j \leq p_j$ and $\bar{p}_j \in \mathbb{N}$ for each $j \in J$ and an assignment of the currently running jobs to the machines, given by a function $\text{run} : \{1, ..., m\} \to J \cup \{\perp\}$ where $\text{run}(i) = \perp$ if at time t no job is running on machine i that was started before t. We allow only configurations in which the jobs in $\text{run}(\{1, ..., m\})$ are exactly the partially processed jobs, i.e, the jobs j with $0 < \bar{p}_j < p_j$. For each time t there is a configuration that corresponds to the status of OPT at time t. The following lemma is an analog of Lemma 5 of the non-preemptive case.

▶ **Lemma 17.** *For each time $t \in \mathbb{N}$ we can compute in time $(mk/\epsilon)^{O(mk^5)}n^{O(1)}$ a set $\mathcal{C}'(t)$ of at most $(mk/\epsilon)^{O(mk^5)}$ configurations such that if $t \in I'_{\text{low}}$ then one of them corresponds to OPT at time t. The set $\mathcal{C}'(T)$ contains only the configuration c_T in which all jobs have completed.*

Our first stage DP has a DP-cell for each tuple (t, c) where $t \in \{0, ..., T\}$ is a time and c is a configuration in $\mathcal{C}'(t)$ according to Lemma 17. A cell (t, c) corresponds to the subproblem of computing a schedule for the interval $[t, T)$ that schedules all jobs j that are pending at time t according to c or that satisfy $r_j > t$. Moreover, if a job j is running on a machine i at time t according to c, then its remaining part has to finish on machine i before another job can be processed on the same machine.

For the base case, the cell (T, c_T) stores an empty schedule. Suppose we are given a cell (t, c) with $t < T$. The reader may imagine that $t \in I'_{\text{low}}$ and that at time t the schedule OPT' is in configuration c. Intuitively, we guess the smallest time $t' > t$ with $t' \in I'_{\text{low}}$ and the configuration c' of OPT' at time t'. Formally, for each time $t' > t$ with $t' \in \mathbb{N}$, and each configuration $c' \in \mathcal{C}'(t')$ we do the following: in case that c and c' are inconsistent we discard the guess. Formally, note that the information about the partially processed jobs according to c and c' imply when those jobs have to be executed during $[t, t')$. We discard the guess if this would imply that a machine has to work on two jobs at the same time, if a job has to be started at different times or on different machines according to c and c', if in c' a job is pending that is already finished according to c, or if the resulting schedule would contradict Lemma 7. In case that $t' = t + 1$ we have that c and c' imply the schedule during $[t, t + 1)$, up to permutation of machines that work on unit size jobs during $[t, t + 1)$. We append to this schedule the schedule stored in the cell (t', c') for the time interval $[t', \infty)$.

If $t' > t+1$ then we create a subproblem for the second stage DP. This subproblem consists of

- the time interval $[t, t') =: [\tau_L, \tau_R)$,
- the set of jobs J' that have to be completely processed during $[\tau_L, \tau_R)$ according to c, and c', i.e., the jobs j that are ready at time τ_L according to c or satisfy $r_j > \tau_L$ and that are finished at time τ_R according to c', and
- the at most $2m$ jobs that are partially but not completely processed at time τ_L and τ_R according to c and c', together with their starting times and the information on which machine they are assigned. Note that since τ_L and τ_R are fixed, there are only $2k$ possibilities for the starting time of each such job. We denote by S the resulting schedule for these jobs.

Denote by (τ_L, τ_R, J', S) the resulting tuple which forms the input to our second stage DP. In this DP, intuitively we scan the time axis from left to right in order to compute a schedule for J' that satisfies the pack-property and that does not conflict with S. If at a time $\tau \in [\tau_L, \tau_R)$ no job class is waiting then we guess the schedule for the interval $[\tau, \tau+1)$ and continue with the time point $\tau+1$. Otherwise, we select a waiting job class $J_{\ell,\ell'}$ of highest density among all waiting job classes and schedule a pack of $J_{\ell,\ell'}$ as soon as possible.

Formally, our second stage DP has an entry (τ, c) for each combination of a time $\tau \in \{\tau_L, ..., \tau_R - 1\}$ and a configuration $c \in \mathcal{C}''(\tau)$ where $\mathcal{C}''(\tau)$ is the set of all configurations of the second stage DP for time τ, consisting only of jobs in J', such that for each job class $J_{\ell,\ell'}$ the total processing time of the ready jobs is less than $4m^2kk!$, i.e., there is no job class that is waiting at time τ. The subproblem for (τ, c) is the problem of computing a schedule S' for the interval $[\tau, \tau_R)$ for the jobs $j \in J'$ that are pending at time τ according to c or which satisfy $r_j > \tau$ and we require that this schedule is consistent with S, i.e., at each time τ no machine schedules one job according to S and another job according to S'. The cost of this schedule is the cost of all jobs in J' that finish during $[\tau, \tau_R)$.

▶ **Lemma 18.** *The number of DP-cells of the second stage DP is bounded by* $(mk)^{O(mk^4)}n^2k^2$.

Suppose we want to solve a subproblem (τ_L, τ_R, J', S) using our second stage DP. For the base case, let $\bar{c}_{\tau_R}$ denote the configuration in which $p'_j = 0$ for each job $j \in J'$. We define $(\tau_R, \bar{c}_{t_R})$ to be the empty schedule, for all other configurations $c'' \in \mathcal{C}''(\tau_R)$ we mark the DP-cell (τ_R, c'') as invalid. Intuitively, a cell (τ, c'') is marked invalid if there is no non-preemptive schedule for $[\tau, \tau_R)$ that finishes before time τ_R all jobs that are have to be processed during this interval according to c''. We describe now how to compute the entry for a cell (τ, c) such that $c \in \mathcal{C}''(\tau)$ and $\tau < \tau_R$. We guess the schedule for the time interval $[\tau, \tau+1)$, i.e., for each machine i that at time τ is not processing a previously started job we try all possibilities to choose a job to start and also allow for the possiblity to keep the machine idle during $[\tau, \tau+1)$. For jobs that start at time τ we require that within their job class they are the pending jobs with earliest release dates (like in OPT', see Lemma 7). Hence, there are only $\tilde{g}(k, \epsilon, m)$ possibilities for some function $\tilde{g}$.

For a given guess, let $\tilde{c}$ denote the resulting configuration at time $\tau+1$. If $\tilde{c} \in \mathcal{C}''(\tau+1)$ then we append the schedule stored in the cell $(\tau+1, \tilde{c})$, unless $(\tau+1, \tilde{c})$ was marked invalid in which case we discard our guess. If $\tilde{c} \notin \mathcal{C}''(\tau+1)$ then according to c' there must be a job class that is waiting. Let τ' be the smallest time step with $\tau+1 \leq \tau'$ such that a machine finishes a job on which it was working at time τ. In this case, let $J_{\ell,\ell'}$ be the job class with highest density that is waiting at time τ', breaking ties like in the construction of OPT'. Let $Q_{\ell,\ell'}$ denote the $mk!/\ell$ ready jobs with earliest release dates among the ready jobs in $J_{\ell,\ell'}$. For each machine i denote by $\tau(i)$ the time when it completes its job that it

executes during $[\tau, \tau+1)$ (according our guess and the jobs that started before τ) and let $\tau(i) := \tau + 1$ if there is no such job. On each machine i we schedule $k!/\ell$ jobs from $Q_{\ell,\ell'}$ during $[\tau(i), \tau(i)+k!)$. Let $\tilde{c}'$ denote the resulting configuration of the pending jobs in J' at time $\tau + 1 + k!$. If $\tilde{c}' \in \mathcal{C}''(\tau + 1 + k!)$ then the solution for our guess of the schedule during $[\tau, \tau+1)$ consists of the schedule for the jobs running at time τ according c, the just computed schedule for the jobs in $Q_{\ell,\ell'}$, and the solution stored in the cell $(\tau + 1 + k!, \tilde{c}')$, unless $(\tau + 1 + k!, \tilde{c}')$ was marked invalid in which case we discard our guess. Otherwise, there must be a waiting job class. In this case we repeat the process (i.e., identify a waiting jobs class and schedule a pack of its jobs as soon as possible) until we arrive at a time τ'' and a configuration $\tilde{c}'' \in \mathcal{C}''(\tau'')$ in which case we append the schedule in the DP-cell $(\tau'', \tilde{c}'')$ to the just computed schedule, or, again discard our guess in case that $(\tau'', \tilde{c}'')$ was marked invalid. In case that the resulting schedule contradicts S (i.e., according to the computed schedule a machine i has to work on a job $j \in J'$ at the same time as it has to work on some job $j' \notin J'$ according to S) we discard our initial guess for the interval $[\tau, \tau+1)$. Finally, in the cell (τ, c) we store the computed schedule of minimum cost over all guesses of the schedule for the interval $[\tau, \tau+1)$, or, in case that we discarded all guesses, we mark the cell (τ, c) as invalid.

Finally, let $\bar{c}_0$ denote the configuration in which $p'_j = p_j$ for each job $j \in J'$, i.e., the configuration of the jobs J' at time τ_L. We observe that $\bar{c}_0 \in \mathcal{C}''(\tau_L)$ since $c \in \mathcal{C}'(\tau_L)$ (where c is the configuration of the cell (t, c) of the first stage DP above) and hence the solution to the subproblem (τ_L, τ_R, J', S) is stored in the cell $(\tau_L, \bar{c}_0)$.

▶ **Theorem 19.** *For the problem of minimizing the weighted flow time on m identical machines with job release dates and without preemption, $P|r_j|\sum w_j F_j$, there is a $(1+\epsilon)$-approximation algorithm with a running time of $(mk/\epsilon)^{O(mk^5)} \cdot n^{O(1)}$, assuming that $p_j, w_j \in \{1, ..., k\}$ for each job j.*

References

1. Foto N. Afrati, Evripidis Bampis, Chandra Chekuri, David R. Karger, Claire Kenyon, Sanjeev Khanna, Ioannis Milis, Maurice Queyranne, Martin Skutella, Clifford Stein, and Maxim Sviridenko. Approximation schemes for minimizing average weighted completion time with release dates. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, pages 32–44. IEEE, 1999.
2. Yossi Azar and Noam Touitou. Improved online algorithm for weighted flow time. *CoRR*, abs/1712.10273, 2017. `arXiv:1712.10273`.
3. Nikhil Bansal. Minimizing flow time on a constant number of machines with preemption. *Oper. Res. Lett.*, 33(3):267–273, 2005. `doi:10.1016/j.orl.2004.07.008`.
4. Nikhil Bansal and Kedar Dhamdhere. Minimizing weighted flow time. *ACM Trans. Algorithms*, 3(4), 2007. `doi:10.1145/1290672.1290676`.
5. Nikhil Bansal and Kirk Pruhs. The geometry of scheduling. *SIAM Journal on Computing*, 43(5):1684–1698, 2014.
6. Jatin Batra, Naveen Garg, and Amit Kumar. Constant factor approximation algorithm for weighted flow time on a single machine in pseudo-polynomial time. *CoRR*, abs/1802.07439, 2018. `arXiv:1802.07439`.
7. Chandra Chekuri and Sanjeev Khanna. A ptas for minimizing weighted completion time on uniformly related machines. In Fernando Orejas, Paul G. Spirakis, and Jan van Leeuwen, editors, *Automata, Languages and Programming*, pages 848–861, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.

8 Chandra Chekuri and Sanjeev Khanna. Approximation schemes for preemptive weighted flow time. In *Proceedings of the thiry-fourth annual ACM Symposium on Theory of computing*, pages 297–305. ACM, 2002.

9 Chandra Chekuri, Sanjeev Khanna, and An Zhu. Algorithms for minimizing weighted flow time. In *Proceedings of the thirty-third annual ACM Symposium on Theory of computing*, pages 84–93. ACM, 2001.

10 Leah Epstein and Asaf Levin. Minimum total weighted completion time: Faster approximation schemes. *CoRR*, abs/1404.1059, 2014. `arXiv:1404.1059`.

11 Naveen Garg, Amit Kumar, and V. N. Muralidhara. Minimizing total flow-time: The unrelated case. In *Proceedings of the 19th International Symposium on Algorithms and Computation (ISAAC 2008)*, pages 424–435. Springer, 2008. `doi:10.1007/978-3-540-92182-0_39`.

12 Fabrizio Grandoni, Stefan Kratsch, and Andreas Wiese. Parameterized approximation schemes for independent set of rectangles and geometric knapsack, 2017. Unpublished.

13 Hans Kellerer, Thomas Tautenhahn, and Gerhard Woeginger. Approximability and nonapproximability results for minimizing total flow time on a single machine. *SIAM Journal on Computing*, 28(4):1155–1166, 1999.

14 Dusan Knop and Martin Koutecký. Scheduling meets n-fold integer programming. *CoRR*, abs/1603.02611, 2016. `arXiv:1603.02611`.

15 Stefano Leonardi and Danny Raz. Approximating total flow time on parallel machines. In *Proceedings of the twenty-ninth annual ACM Symposium on Theory of Computing*, pages 110–119. ACM, 1997.

16 Joseph Y-T. Leung. *Handbook of scheduling: algorithms, models, and performance analysis*. CRC Press, 2004.

17 Dániel Marx. Parameterized complexity and approximation algorithms. *The Computer Journal*, 51(1):60–78, 2008.

18 Matthias Mnich and Andreas Wiese. Scheduling and fixed-parameter tractability. *Math. Program.*, 154(1-2):533–562, 2015.

19 Andreas Wiese. A (1+epsilon)-approximation for unsplittable flow on a path in fixed-parameter running time. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017*, pages 67:1–67:13, 2017.

A Omitted proofs

A.1 Proof of Lemma 4

We define a partially fractional objective function: for a given solution S and a value $t \in \Gamma$ let $\tilde{W}_S(t) = \sum_{j \in J: r_j \leq t} \frac{w_j}{p_j} p_j^S(t)$ where $p_j^S(t)$ denotes the total length of each job j that has not been finished by time t in S. We define a new objective function by $\sum_{t \in I_{\text{low}} \cap \Gamma} \epsilon W(t) + \sum_{t \in I_{\text{high}} \cap \Gamma} \epsilon \tilde{W}(t)$. We have that $\sum_{t \in I_{\text{low}} \cap \Gamma} \epsilon W_{OPT'}(t) + \sum_{t \in I_{\text{high}} \cap \Gamma} \epsilon \tilde{W}_{OPT'}(t) \leq \sum_{t \in I_{\text{low}} \cap \Gamma} \epsilon W_{OPT}(t) + \sum_{t \in I_{\text{high}} \cap \Gamma} \epsilon \tilde{W}_{OPT}(t)$ since the solution values of our fractional schedules in the maximally large intervals of I_{high} are a lower bound on the respective cost of OPT in these intervals.

On the other hand, for each $t \in I_{\text{high}} \cap \Gamma$ we have that $W_{OPT'}(t) \leq \tilde{W}_{OPT'}(t) + 3mk^3 \leq \tilde{W}_{OPT'}(t) + \epsilon W_{OPT}(t)$ since $W_{OPT}(t) > 3mk^3/\epsilon^2$ and for each job class and each time t there are at most $3mk/\epsilon$ jobs that have been started in OPT' but not yet completed: at most mk/ϵ jobs j for which $0 < \bar{p}_j(t) < p'_j$ (where p'_j denotes the amount of work that OPT finishes on job j during the maximally large subinterval of I_{high} that contains t), at most mk/ϵ jobs j with $\bar{p}_j(t) = p'_j$ but $p'_j < p_j$ (those jobs are partially scheduled at time t_2 in OPT and by Lemma 2 there can be only mk/ϵ of those) and at most mk/ϵ jobs j with

$\bar{p}_j(t) = 0$ but that have already been partially processed in OPT at time t_1 (again, due to Lemma 2). This yields

$$\begin{aligned}\int_t W_{OPT'}(t)dt &= \sum_{t\in\Gamma} \epsilon W_{OPT'}(t)\\ &\leq \sum_{t\in\Gamma\cap I_{\text{low}}} \epsilon W_{OPT'}(t) + \sum_{t\in\Gamma\cap I_{\text{low}}} \left(\epsilon\tilde{W}_{OPT'}(t) + \epsilon^2\cdot W_{OPT}(t)\right)\\ &\leq \sum_{t\in\Gamma\cap I_{\text{low}}} \epsilon W_{OPT'}(t) + \sum_{t\in\Gamma\cap I_{\text{low}}} \left(\epsilon W_{OPT}(t) + \epsilon^2\cdot W_{OPT}(t)\right)\\ &\leq c(OPT) + \epsilon\cdot c(OPT)\\ &\leq (1+\epsilon)c(OPT).\end{aligned}$$

A.2 Proof of Lemma 9

We will transform OPT step by step into a schedule for J' satisfying the pack-property. We consider the timesteps $t \in \{t_1, ..., t_2 - 1\}$ in increasing order. Define $OPT_{t_1-1} := OPT$. For each timestep $t \in \{t_1, ..., t_2 - 1\}$ we take the schedule OPT_{t-1}, assuming inductively that $OPT_{t'}$ fulfills the pack-property for each $t' \in \{t_1, ..., t-1\}$. If in OPT_{t-1} at time t there is no waiting job class $J_{\ell,\ell'}$ then we define $OPT_t := OPT_{t-1}$ and hence OPT_t fulfills the pack-property at each $t' \in \{t_1, ..., t\}$. Otherwise, let $J_{\ell,\ell'}$ denote a waiting job class with maximum density ℓ'/ℓ, breaking ties in an arbitrary fixed order. Denote by $\tilde{J}_{\ell,\ell'}$ the at least $4m^2k^2k!$ jobs from $J_{\ell,\ell'}$ that are ready at time t. Hence, there must be a machine i^* such that jobs from $\tilde{J}_{\ell,\ell'}$ with a total processing time of $4mkk!$ are scheduled on i^* during $[t, t_2)$. Note that this implies in particular that $t_2 \geq t + 4m^2k^2k!$. For i let $t(i)$ be the earliest time $t' \geq t$ when machine i starts a new job or is idle. We change the order of the jobs on i^* such that after $t(i^*)$ we first schedule the jobs in $\tilde{J}_{\ell,\ell'}$ on i^* and then all other jobs that are scheduled on i^* in OPT_{t-1} in the same order as in OPT_{t-1}. Next, we swap some of the jobs in $\tilde{J}_{\ell,\ell'}$ on i^* to the other machines such that each machine i schedules $k!/\ell$ jobs from $\tilde{J}_{\ell,\ell'}$.

▶ **Claim 20.** *For each machine $i \neq i^*$ there is a set of jobs $J(i)$ that i starts and completes during $[t, t + 2k\cdot k! + k)$ such that there are $k!$ time units in $[t, t + 2k\cdot k!)$ during which machine i either works on a job in $J(i)$ or is idle.*

Proof. It suffices to define $J(i) := \emptyset$ if the total amount of idle time on machine i during $[t(i), t(i) + 2k\cdot k!)$ is at least $k!$. Otherwise, during $[t(i), t(i) + 2k\cdot k! + k)$ machine i starts and completes jobs with a total processing time of at least $kk!$ and hence there must be an integer $k' \leq k$ such that i starts and completes at least $k!/k'$ jobs of size exactly k'. We define $J(i)$ to be $k!/k'$ of these jobs. ◀

Since machine i^* schedules jobs from $\tilde{J}_{\ell,\ell'}$ with a total processing time of at least $4mkk! \geq mk! + 2kk! + k$, there must be $mk!/\ell$ of them which start at time $t + 2kk! + k$ or later. For each machine $i \neq i^*$ we swap the jobs $J(i)$ with $k!/\ell$ jobs on i^* from $\tilde{J}_{\ell,\ell'}$ that start at time $t + 2k\cdot k! + k$ or later. On machine i, after time $t(i)$ we first schedule the jobs that were previously on i^* and then schedule the remaining jobs in the same order as in OPT_{t-1}. No job is started before its release date since all jobs in $\tilde{J}_{\ell,\ell'}$ are ready at time t. Also, since in OPT_{t-1} there are $k!$ time units during $[t, t + 2k\cdot k! + k)$ in which i either works on a job in $J(i)$ or is idle, each job in J' on i completes by time $t_2(i)$ the latest. On i^* we first schedule all jobs from $\tilde{J}_{\ell,\ell'}$ and then the jobs from the sets $J(i)$ in an arbitrary order. Since each job in $J(i)$ starts before time $t + 2k\cdot k! + k$ in OPT_{t-1} it starts no earlier than its release date (after the swap).

A.3 Proof of Lemma 14

Based on OPT, we construct a schedule $\overline{OPT}$ for the jobs in J_{p} that fulfills that $W_{\overline{OPT}}(t) \leq W_{OPT}(t) + 4m^2k^5k!$ for each t and in which no job j in a pack Q_r is executed before time t'_r. Then the claim of the lemma follows since for each such schedule S and each t we have that $\tilde{W}_{FRAC}(t) \leq W_S(t)$. Consider a job class $J_{\ell,\ell'}$ and let $Q_1, Q_2, ...$ be its corresponding packs and let $t'_1, t'_2, ...$ be there corresponding times t'. To construct $\overline{OPT}$, for each pack Q_s we schedule the jobs in Q_s during the times when OPT schedules the jobs in $Q_{s+4m^2k^2}$. Observe that for each job $j \in Q_{s+4m^2k^2}$ it holds that $r_j \geq t'_s$ and hence after this swap each job in Q_s is started no earlier than t'_s. We remove the jobs in the last $4m^2k^2$ packs of $J_{\ell,\ell'}$ from the schedule, one may imagine that we schedule them at the very end after t_2. So intuitively, at each time t the schedule $\overline{OPT}$ is at most $4m^2k^2$ packs "behind" the schedule OPT. The total weight of the jobs in $4m^2k^2$ packs is bounded by $k \cdot 4m^2k^2k!$. This implies that $W^{J_{\ell,\ell'}}_{\overline{OPT}}(t) \leq W^{J_{\ell,\ell'}}_{OPT}(t) + 4m^2k^3k!$ for each t where $W^{J_{\ell,\ell'}}_{OPT}(t)$ and $W^{J_{\ell,\ell'}}_{\overline{OPT}}(t)$ denote the pending weight of the jobs in $J_{\ell,\ell'}$ at time t, respectively. Since there are at most k^2 jobs classes, this implies that $W_{\overline{OPT}}(t) \leq W_{OPT}(t) + 4m^2k^5k!$ and hence $\tilde{W}_{FRAC}(t) \leq W_{OPT}(t) + 4m^2k^5k!$.

A.4 Proof of Lemma 15

The second inequality follows from the first one by a geometric sum argument. We prove the first inequality. The claim is clearly true for $t = t_1$. For all other t we prove the lemma by induction over the groups $J^{(1)}_{\mathrm{p}}, J^{(2)}_{\mathrm{p}}, ..., J^{(\gamma)}_{\mathrm{p}}$. For the base case, consider the group $J^{(\gamma)}_{\mathrm{p}}$. The jobs in $J^{(\gamma)}_{\mathrm{p}}$ have the maximum density among all jobs in the instance. For this group, we prove the claim by induction over t. More precisely, we prove that $W^{J^{(\gamma)}_{\mathrm{p}}}_{OPT'_{\mathrm{p}}}(t) \leq \tilde{W}^{J^{(\gamma)}_{\mathrm{p}}}_{FRAC}(t) + m \cdot k!$ for each t. The claim is true for $t = t_1$. Consider an arbitrary time t and suppose that the claim is true for each time $t' < t$. If during $[t-1, t)$ in the schedule OPT'_{p} one machine is working on a job in $J^{(\gamma)}_{\mathrm{p}}$ then all machines are working on a job in $J^{(\gamma)}_{\mathrm{p}}$ and thus the claim is also true for time t. Similarly, if during $[t-1, t)$ the schedule $FRAC$ is not working on a job in $J^{(\gamma)}_{\mathrm{p}}$ then the claim is also true for time t. Otherwise, assume that during $[t-1, t)$ in the schedule OPT'_{p} no machine is working on a job in $J^{(\gamma)}_{\mathrm{p}}$ but in $FRAC$ some machine is working on a job in $J^{(\gamma)}_{\mathrm{p}}$. Hence, in $FRAC$ all machines are working on a job in $J^{(\gamma)}_{\mathrm{p}}$ during $[t-1, t)$. Thus, there is a pack Q_r consisting of jobs in $J^{(\gamma)}_{\mathrm{p}}$ with $t_r \leq t-1$ of density γ with $t_r \leq t-1$ but OPT'_{p} does not yet work on Q_r. This must be because OPT'_{p} has already started processing jobs from another pack $Q_{r'}$ but it has not yet finished them. Suppose that OPT'_{p} started the pack $Q_{r'}$ at a time $\hat{t} < t-1$. Then, however, $\hat{t} \geq t - 1 - (k! - 1) = t - k!$ and $W^{J^{(\gamma)}_{\mathrm{p}}}_{OPT'_{\mathrm{p}}}(\hat{t}) = 0$. The schedule $FRAC$ processes the jobs in $J^{(\gamma)}_{\mathrm{p}}$ at least as fast as OPT'_{p} and thus also $\tilde{W}^{J^{(\gamma)}_{\mathrm{p}}}_{FRAC}(\hat{t}) = 0$. The schedule $FRAC$ can schedule at most one job from $J^{(\gamma)}_{\mathrm{p}}$ at a time on each machine. Hence, $W^{J^{(\gamma)}_{\mathrm{p}}}_{OPT'_{\mathrm{p}}}(t) \leq \tilde{W}^{J^{(\gamma)}_{\mathrm{p}}}_{FRAC}(t) + mk \cdot k! = \tilde{W}^{J^{(\gamma)}_{\mathrm{p}}}_{FRAC}(t) + 2^{\gamma-r}mk \cdot k!$.

Now assume that there is a value g such that the claim is true for each $g' > g$. Consider the group $J^{(g)}_{\mathrm{p}}$. Again, we prove the claim for $J^{(g)}_{\mathrm{p}}$ by induction over t. At time $t = t_1$ the claim is clearly true. Suppose that there is a time t such that for each $t' < t$ the claim is true. The claim is immediately true for t if in OPT'_{p} at least one machine (and hence all machines) work on a job of group $J^{(g)}_{\mathrm{p}}$ during $[t-1, t)$ or if in $FRAC$ no machine is working on a job in $J^{(g)}_{\mathrm{p}}$ during $[t-1, t)$. Therefore, let us assume that the latter statements are both not true. Let $t'' \leq t$ be the latest point in time before t when $W^{J^{(g)}_{\mathrm{p}}}_{OPT'_{\mathrm{p}}}(t'') \leq \tilde{W}^{J^{(g)}_{\mathrm{p}}}_{FRAC}(t'')$. If $t'' = t$ there is nothing to show so assume that $t'' < t$. Hence, during $[t'', t''+1)$ in the

schedule $FRAC$ all machines work on a job in $J_{\mathrm{p}}^{(g)}$ but in OPT'_{p} no machine works on a job in $J_{\mathrm{p}}^{(g)}$. In particular, at time t'' there is a pack Q_r with jobs in $J_{\mathrm{p}}^{(g)}$ with $t'_r \leq t''$ but at time t'' it has neither been completely processed by OPT'_{p} nor by $FRAC$. Also, OPT'_{p} does not process jobs from Q_r during $[t'', t''+1)$ (otherwise this would contradict our definition of t'') and hence Q_r has not been started in OPT'_{p}. Since OPT'_{p} and $FRAC$ order the packs from each group $J_{\mathrm{p}}^{(g')}$ according to the artificial release dates t'_r this implies that also in $FRAC$ subpack Q_r has not been started yet.

Also, at time t'' in $FRAC$ there is no job available from a group $J_{\mathrm{p}}^{(g')}$ with $g' > g$ and thus $\tilde{W}_{FRAC}^{J_{\mathrm{p}}^{(g')}}(t'') = 0$ for each such g'. Hence, the induction hypothesis implies that $\tilde{W}_{OPT'_{\mathrm{p}}}^{J_{\mathrm{p}}^{(g')}}(t'') < 2^{\gamma-g'} m \cdot k!$ for each such g'. Let $\bar{J}$ denote all jobs in packs $Q_{r'}$ with $t'_{r'} \in [t'', t)$ and which are contained in a group $J_{\mathrm{p}}^{(g')}$ with $g' > g$. Since $FRAC$ works on a jobs in $J_{\mathrm{p}}^{(g)}$ during $[t-1, t)$ all jobs in $\bar{J}$ finish in $FRAC$ during $[t'', t)$. Hence, during $[t'', t)$ in the schedule $FRAC$ there are at most $t - t'' - p(\bar{J})/m$ units of time during which at least one (and hence all) machines work on jobs in a group $J_{\mathrm{p}}^{(g')}$ with $g' > g$. At each time $\hat{t} \in [t'', t)$ we have that $W_{OPT'_{\mathrm{p}}}^{J_{\mathrm{p}}^{(g)}}(\hat{t}) \geq 1$ since otherwise this would contradict our choice of t''. Hence, if at a time $\hat{t} \in [t'', t)$ the schedule OPT'_{p} does not work on a job in $J_{\mathrm{p}}^{(g)}$ then OPT'_{p} works on a job in a pack that is partially processed at time t'' or on a job from a group $J_{\mathrm{p}}^{(g')}$ with $g' > g$. Recall that $\tilde{W}_{OPT'_{\mathrm{p}}}^{J_{\mathrm{p}}^{(g')}}(t'') < 2^{\gamma-g'} m \cdot k!$ for each $g' > g$. Hence, the total time, summed up over all machines, during $[t'', t)$ when OPT'_{p} does not work on a job in $J_{\mathrm{p}}^{(g)}$ is bounded by $m \cdot k! + p(\bar{J}) + \sum_{g':g'>g} 2^{\gamma-g'} m \cdot k!$. Therefore, OPT'_{p} finishes at least $m(t-t'') - p(\bar{J}) - m \cdot k! - \sum_{g':g'>g} 2^{\gamma-g'} m \cdot k!$ units of work of jobs in $J_{\mathrm{p}}^{(g)}$ during $[t'', t)$. Since $W_{OPT'_{\mathrm{p}}}^{J_{\mathrm{p}}^{(g)}}(t'') \leq \tilde{W}_{FRAC}^{J_{\mathrm{p}}^{(g)}}(t'')$ we have that $W_{OPT'_{\mathrm{p}}}^{J_{\mathrm{p}}^{(g)}}(t) - \tilde{W}_{FRAC}^{J_{\mathrm{p}}^{(g)}}(t) \leq m(t-t'') - p(\bar{J}) - (m(t-t'') - p(\bar{J}) - m \cdot k! - \sum_{g':g'>g} 2^{\gamma-g'} m \cdot k!) = 2^{\gamma-g} m \cdot k!$ and therefore $W_{OPT'_{\mathrm{p}}}^{J_{\mathrm{p}}^{(g)}}(t) \leq \tilde{W}_{FRAC}^{J_{\mathrm{p}}^{(g)}}(t) + 2^{\gamma-g} m \cdot k!$.

A.5 Proof of Lemma 16

We first prove that for each time $t \in I'_{\mathrm{high}}$ it holds that $W_{OPT'}(t) \leq (1+\epsilon) W_{OPT}(t)$. Let $t \in I'_{\mathrm{high}}$. We have that

$$
\begin{aligned}
W_{OPT'}(t) &= W^{\mathrm{np}}_{OPT'}(t) + W^{\mathrm{p}}_{OPT'}(t) \\
&\overset{\text{Proposition 10}}{\leq} 4m^2k^4k! + W^{\mathrm{p}}_{OPT'}(t) \\
&\overset{\text{Lemma 12}}{\leq} 4m^2k^4k! + W_{OPT''_{\mathrm{p}}}(t) + k^2 m \\
&\leq 5m^2k^4k! + \sum_{g=1}^{\gamma} W_{OPT''_{\mathrm{p}}}^{J_{\mathrm{p}}^{(g)}}(t) \\
&\overset{\text{Lemma 15}}{\leq} 5m^2k^4k! + \sum_{g=1}^{\gamma} \left(\tilde{W}_{FRAC}^{J_{\mathrm{p}}^{(g)}}(t) + 2^{\gamma-g} m \cdot k! \right) \\
&\leq 5m^2k^4k! + 2^{\gamma} m \cdot k! + \tilde{W}_{FRAC}(t) \\
&\overset{\text{Lemma 14}}{\leq} 5m^2k^4k! + 2^{k^2} m^2 \cdot k! + W_{OPT}(t) + 4m^2k^5k! \\
&\leq \epsilon W_{OPT}(t) + W_{OPT}(t) \\
&= (1+\epsilon) W_{OPT}(t).
\end{aligned}
$$

Since $W_{OPT'}(t) = W_{OPT}(t)$ for each $t \in I'_{\text{low}}$ we conclude that $c(OPT') = \sum_t W_{OPT'}(t) \leq (1+\epsilon)\sum_t W_{OPT}(t) = (1+\epsilon)c(OPT)$.

A.6 Proof of Lemma 17

Since $W_{OPT}(t) \leq 2^{2k^2}m^2/\epsilon$ for each $t \in I'_{\text{low}}$ there can be at most $2^{2k^2}m^2/\epsilon$ pending jobs at time t. For each of them, there are k^2 options to which class it belongs to and k options for its remaining processing time. For each job class, there are $m!$ possibilities for the identities of the at most m partially scheduled jobs and additionally $m!$ possibilities for the machines on which the partially processed jobs run. This yields $\left(1 + 2^{2k^2}m^2/\epsilon\right)^{k^3} (m!)^{k^2+1} = (mk/\epsilon)^{O(mk^5)}$ possible configurations for the pending jobs, which of them are running at time t on the machines, and how much of them has already been processed.

A.7 Proof of Lemma 18

For each job class there can be at most $4m^2kk!$ pending jobs and there are k options for their remaining processing time. This yields $(1 + 4m^2kk!)^k$ options per job class. There are k^2 job classes and there are $m!$ possibilities to distribute the partially processed jobs on the m machines. This yields $m!(1 + 4m^2kk!)^{k^3} = (mk)^{O(mk^4)}$ options in total for each $\tau \in \{\tau_L, ..., \tau_R\}$.

A.8 Proof of Theorem 19

We first prove the correctness of the second stage DP and then based on this the correctness of the first stage DP. Given an instance (τ_L, τ_R, J', S) of the second stage DP such that $\tau_L, \tau_R \in I'_{\text{low}}$, S is the schedule of all jobs that are partially processed at times τ_L or τ_R in OPT', and J' is the set of jobs that OPT' completely finishes during $[\tau_L, \tau_R)$. We prove that the second stage DP computes a schedule S' for the interval $[\tau_L, \tau_R)$ in which

- all jobs in J' are completed
- at each time $\tau \in [\tau_L, \tau_R)$ each machine i either works on a job $j \in J'$, or works on some job $j \notin J'$ according to S, or works on no job at all, and
- the cost of the jobs J' in S' is bounded by the cost of the jobs in J' in OPT', i.e., $\sum_{j\in J'} w_j(C_j^{S'} - r_j) \leq \sum_{j\in J'} w_j(C_j^{OPT'} - r_j)$ where $C_j^{S'}$ and $C_j^{OPT'}$ denote the completion times of job j in S' and OPT', respectively.

In this proof, whenever we refer to a configuration c of OPT' at a time τ, we mean the configuration of OPT' restricted to the jobs in J'. We prove by induction for each cell (τ, c) of the second stage DP such that OPT' is in configuration c at time τ and $c \in \mathcal{C}''(\tau)$ that the cost of the schedule for the interval $[\tau, \tau_R)$ stored in (τ, c) is upper-bounded by the cost of OPT' during $[\tau, \tau_R)$. The claim is true by definition for the base case where $\tau = \tau_R$ since then both schedules are empty. Suppose that there is a value $\tau \in [\tau_L, \tau_R)$ such that the claim is true for each (τ', c') with $\tau' \in [\tau_L, \tau_R)$ and $\tau' > \tau$ such that $c' \in \mathcal{C}''(\tau')$ and OPT' is in configuration c' at time τ'. We show that the claim is true for the cell (τ, c) such that OPT' is in configuration c at time τ in case that $c \in \mathcal{C}''(\tau)$. Let c be the configuration of OPT' at time τ. In case that $c \notin \mathcal{C}''(\tau)$ there is nothing to show. Assume that $c \in \mathcal{C}''(\tau)$. Among all our guesses for the schedule of the interval $[\tau, \tau+1)$ we enumerate one guess that coincides with the schedule of OPT' during $[\tau, \tau+1)$. Let c' denote the resulting configuration at time $\tau+1$. If $c' \in \mathcal{C}''(\tau+1)$ then by induction the schedule corresponding to this guess (i.e., the guessed schedule for $[\tau, \tau+1)$ plus the schedule for $[\tau+1, \tau_R)$ stored in the cell $(\tau+1, c')$) has a cost of at most the cost of OPT' during $[\tau, \tau_R)$, i.e., the cost of the jobs finishing during

$[\tau, \tau_R)$ in OPT'. If $c' \notin \mathcal{C}''(\tau+1)$ then we schedule a pack or several packs of jobs until we reach a time τ'' and a configuration $\tilde{c}'' \in \mathcal{C}''(\tau'')$. Since OPT' satisfies the pack property, we produce a schedule satisfying the pack property, and at each time $\hat{\tau} \in \{\tau+1, ..., \tau''-1\}$ there is a waiting job class or all machines are busy with pack jobs, the resulting schedule for the interval $[\tau, \tau'')$ is identical to the schedule of OPT' during $[\tau, \tau'')$. Hence, by induction we know that the resulting schedule for the interval $[\tau, \tau_R)$ for this guess has a cost that is at most the cost of OPT' during the interval $[\tau, \tau_R)$. This completes the proof for the second stage DP.

For the first stage DP, one can prove correctness with a similar induction as in the proof of Theorem 6, using that for a cell (t, c) with $t+1 \notin I'_{\text{low}}$ the DP guesses the earliest time t' with $t' > t$ and $t' \in I'_{\text{low}}$, the configuration c' of OPT' at time t', and that then the second stage DP computes a schedule for the interval $[t, t')$ which completes the same set of jobs that OPT' completes during $[t, t')$ and whose cost is upper-bounded by the cost of OPT' for these jobs.

It remains to prove the running time of the algorithm. The number of DP cells of the first stage DP is bounded by $(mk/\epsilon)^{O(mk^5)}O(n^2k^2)$, using Lemma 17 and that $T \leq O(n^2k^2)$. In order to compute the entry of a cell (t, c) of the first stage DP, we enumerate over $(mk/\epsilon)^{O(mk^5)}n^2k^2$ possibilities for (t', c'). In case that $t' = t+1$ we can compute the schedule for $[t, t+1)$ in time $n^{O(1)}$.

If $t' > t+1$ we additionally solve an instance of the second stage DP. The number of DP-cells of the second stage DP is bounded by $(mk)^{O(mk^4)}n^2k^2$. To compute the entry of one cell (t, c) we enumerate over $k^{O(m)}$ schedules for the interval $[t, t+1)$ and then possibly compute a schedule for pack jobs in time $n^{O(1)}$. Hence, we solve the resulting instance of the second stage DP in time $(mk)^{O(mk^4)}n^2k^2k^{O(m)}n^{O(1)} = (mk)^{O(mk^4)}n^{O(1)}$. Therefore, the schedule of a DP cell of the first stage DP can be computed in time $(mk/\epsilon)^{O(mk^5)}n^2k^2 \cdot (mk)^{O(mk^4)}n^{O(1)} = (mk/\epsilon)^{O(mk^5)}n^{O(1)}$ and thus the entries of all cells can be computed in time $(mk/\epsilon)^{O(mk^5)}n^{O(1)}(mk/\epsilon)^{O(mk^5)}n^{O(1)} = (mk/\epsilon)^{O(mk^5)}n^{O(1)}$.

List-Decoding Homomorphism Codes with Arbitrary Codomains

László Babai[1]
University of Chicago, Chicago IL, USA
laci@cs.uchicago.edu
https://orcid.org/0000-0002-2058-685X

Timothy J. F. Black[2]
University of Chicago, Chicago IL, USA
timblack@math.uchicago.edu
https://orcid.org/0000-0003-2469-9867

Angela Wuu[3]
University of Chicago, Chicago IL, USA
wu@math.uchicago.edu

Abstract

The codewords of the *homomorphism code* $\mathrm{aHom}(G, H)$ are the affine homomorphisms between two finite groups, G and H, generalizing Hadamard codes. Following the work of Goldreich–Levin (1989), Grigorescu et al. (2006), Dinur et al. (2008), and Guo and Sudan (2014), we further expand the range of groups for which local list-decoding is possible up to $\mathsf{mindist}$, the minimum distance of the code. In particular, for the first time, we **do not require either G or H to be solvable**. Specifically, we demonstrate a $\mathrm{poly}(1/\varepsilon)$ bound on the list size, i. e., on the number of codewords within distance $(\mathsf{mindist} - \varepsilon)$ from any received word, when G is either abelian or an alternating group, and H **is an arbitrary (finite or infinite) group**. We conjecture that a similar bound holds for all finite simple groups as domains; the alternating groups serve as the first test case.

The abelian vs. arbitrary result permits us to adapt previous techniques to obtain efficient local list-decoding for this case. We also obtain efficient local list-decoding for the permutation representations of alternating groups (the codomain is a symmetric group) under the restriction that the domain $G = A_n$ is paired with codomain $H = S_m$ satisfying $m < 2^{n-1}/\sqrt{n}$.

The limitations on the codomain in the latter case arise from severe technical difficulties stemming from the need to solve the *homomorphism extension* (HomExt) *problem* in certain cases; these are addressed in a separate paper (Wuu 2018).

We introduce an intermediate "semi-algorithmic" model we call **Certificate List-Decoding** that bypasses the HomExt bottleneck and works in the alternating vs. arbitrary setting. A certificate list-decoder produces partial homomorphisms that uniquely extend to the homomorphisms in the list. A homomorphism extender applied to a list of certificates yields the desired list.

2012 ACM Subject Classification Mathematics of computing → Coding theory, Mathematics of computing → Probabilistic algorithms

Keywords and phrases Error-correcting codes, Local algorithms, Local list-decoding, Finite groups, Homomorphism codes

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.29

Related Version A full version of this paper appears on arXiv [4], `https://arxiv.org/abs/1806.02969`.

[1] Partially supported by NSF Grants CCF 1423309 and CCF 1718902. The views expressed in the paper are those of the authors and do not necessarily reflect the views of the NSF.
[2] Partially supported by L. Babai's cited NSF grants.
[3] Partially supported by L. Babai's cited NSF grants.

1 Introduction

1.1 Brief history

Let G and H be finite groups, to be referred to as the *domain* and the *codomain*, respectively. A map $\psi\colon G \to H$ is an **affine homomorphism** if

$$(\forall a, b, c \in G)(\psi(a)\psi(b)^{-1}\psi(c) = \psi(ab^{-1}c))\,. \tag{1}$$

Equivalently, ψ is a translate of a homomorphism, i. e., there exists a homomorphism $\varphi\colon G \to H$ and an element $h \in H$ such that $(\forall g \in G)(\psi(g) = \varphi(g) \cdot h)$. We write $\mathrm{Hom}(G, H)$ and $\mathrm{aHom}(G, H)$ to denote the set of homomorphisms and affine homomorphisms, respectively. Let H^G denote the set of all functions $f\colon G \to H$. We represent an (affine) homomorphism ψ by the list of pairs $(g, \psi(g))$ for $g \in S$ where S is a set of (affine) generators of G.

We view $\mathrm{aHom}(G, H)$ as a (nonlinear) code within the code space H^G (the space of possible "received words") and refer to this class of codes as *homomorphism codes*.

Homomorphism codes are candidates for efficient *local* list-decoding *up to minimum distance* (mindist) and in many cases it is known that their minimum distance is (asymptotically) equal to the list-decoding bound.

This line of work goes back to the celebrated paper by Goldreich and Levin (1989) [12] who found local list-decoders for Hadamard codes, i. e., for homomorphism codes with domain $G = \mathbb{Z}_2^n$ and codomain $H = \mathbb{Z}_2$. This result was extended to homomorphism codes of abelian groups (both the domain and the codomain abelian) by Grigorescu, Kopparty, and Sudan (2006) [14] and Dinur, Grigorescu, Kopparty, and Sudan (2008) [10] and to the case of supersolvable domain and nilpotent codomain by Guo and Sudan (2014) [16], cf. [9].

While homomorphism codes have low (logarithmic or worse) rates, they tend to have remarkable list-decoding properties. In particular, in all cases studied so far (including the present paper), for an *arbitrary* received word $f \in H^G$, and any $\varepsilon > 0$, the number of codewords within radius $(\mathsf{mindist} - \varepsilon)$ is bounded by $\mathrm{poly}(1/\varepsilon)$ (as opposed to some faster-growing function of ε, as permitted in the theory of list-decoding). This is an essential feature for the complexity-theoretic application (hard-core predicates) by Goldreich and Levin. Let $\mathcal{L}$ denote the list of codewords within distance $(\mathsf{mindist} - \varepsilon)$ of the received word.

We call an $|\mathcal{L}| \le \mathrm{poly}(1/\varepsilon)$ bound *economical*, and a class of homomorphism codes permitting such a bound **combinatorially economically list-decodable (CombEcon)**. (With some abuse of the language, we shall talk about "a CombEcon code" in reference to members of a class of codes defined by the context. We apply the analogous convention to other asymptotic properties of classes of codes to be defined below as well.)

By *efficient* list-decoding we mean performing $\mathrm{poly}(\log|G|, 1/\varepsilon)$ randomized queries to the received word and performing $\mathrm{poly}(\log|G|, \log|H|, 1/\varepsilon)$ additional work to produce a list of $\le \mathrm{poly}(1/\varepsilon)$ affine homomorphisms that includes all affine homomorphisms within $(\mathsf{mindist} - \varepsilon)$ of the received word.

We call a CombEcon code **AlgEcon (algorithmically economically list-decodable)** if it permits efficient decoding within radius $(\mathsf{mindist} - \varepsilon)$ in this sense. So the cited results show that homomorphism codes with abelian domain and codomain, and more generally with supersolvable domain and nilpotent codomain, are CombEcon and AlgEcon.

In all work on the subject, this efficiency depends on the computational representation of the groups used (presentation in terms of generators and relators, black-box access, black-box groups, permutation groups, matrix groups, etc.). We shall make the representation required explicit in all algorithmic results.

1.2 Our contributions

1.2.1 Combinatorial bounds

In this paper we further expand the range of groups for which efficient local list-decoding is possible up to the minimum distance. In particular, for the first time, we **do not require either G or H to be solvable**. In fact, in our combinatorial and semi-algorithmic results (see below), **the codomain is an arbitrary (finite or infinite) group**. We say that a class $\mathfrak{G}$ of finite groups is **universally CombEcon** if for all $G \in \mathfrak{G}$ and arbitrary (finite or infinite) H, the code $\mathrm{aHom}(G, H)$ is CombEcon. This paper is the first to demonstrate the existence of significant universally CombEcon classes.

▶ **Convention 1.** When speaking of a homomorphism code $\mathrm{aHom}(G, H)$, the domain G will always be a finite group, but the codomain H will, in general, not be restricted to be finite.

▶ **Theorem 2** (Main combinatorial result)**.** *Finite abelian and alternating groups are universally CombEcon.*

We explain this result in detail. By *distance* we mean normalized Hamming distance.

(Restatement of Theorem 2.) *Let the domain G be a finite abelian or alternating group and H an arbitrary (finite or infinite) group. Let* mindist *denote the minimum distance of the homomorphism code* $\mathrm{aHom}(G, H)$ *and let $\varepsilon > 0$. Let $f \in H^G$ be an arbitrary received word. Then the number of codewords within $(\mathsf{mindist} - \varepsilon)$ of f is at most* $\mathrm{poly}(1/\varepsilon)$.

▶ Remark. We give two proofs of this result. The first proof is nonconstructive and is based on a broadly applicable sphere packing argument (see Sec. 3.2). The second proof is more closely based on the structure of the alternating groups and depends on a result about random generation with extremely high probability (see Theorem 20). This approach yields a very simple semi-algorithmic result (certificate list-decoding, see Sec. 1.2.3) and leads, using deeper tools [21], to our main algorithmic result, Theorem 6.

For abelian domains we prove a bound of $O(\varepsilon^{-C-5})$ on the length of the list, i. e., the number of codewords within $(\mathsf{mindist} - \varepsilon)$ of the received word where $O(\varepsilon^{-C})$ is the degree in the corresponding {abelian $\to$ abelian} bound. (Currently $C \approx 105$ [16].) For alternating domains we prove a bound of $\widetilde{O}(\varepsilon^{-7})$ on the length of the list.

Our choice of the alternating groups as the domain is our test case of what we believe is a general phenomenon valid for all finite simple groups.

▶ **Conjecture 3.** *The class of finite simple groups is universally CombEcon.*

The following problem is also open.

▶ **Problem 4.** *Is the class of all finite groups universally CombEcon?*

We suspect the answer to be negative.

Let us say that the **depth** of a subgroup M in a group G is the length ℓ of the longest subgroup chain $M = M_0 < M_1 < \cdots < M_\ell = G$. We say that a subgroup is **shallow** if it has bounded depth.

Theorem 2 also holds for a hierarchy of wider classes of finite groups we call *shallow random generation* groups or "SRG groups." This hierarchy includes the class of alternating groups. The defining feature of SRG groups is that a bounded number of random elements generate, with extremely high probability, a shallow subgroup.

Our combinatorial tools allow us to play on the relatively well-understood top layers of the subgroup lattice of the (alternating or SRG) domain, avoiding the dependence on the codomain in the combinatorial and semi-algorithmic context.

▶ Remark. Our results list-decode certain classes of codes up to distance ($\mathsf{mindist} - \varepsilon$) for positive ε. In many cases, mindist is the list-decoding boundary; examples show that the length of the list may blow up when ε is set to zero. Classes of such examples with abelian domain and codomain were found by Guo and Sudan [16]. We add classes of examples with alternating domains (see Appendix B).

1.2.2 Algorithms

On the algorithmic front, the combinatorial bound in the {abelian → arbitrary} case permits us to adapt the algorithm of [14] to obtain efficient local list-decoding. We say that a class $\mathfrak{G}$ of finite groups is **universally AlgEcon** if for all $G \in \mathfrak{G}$ and arbitrary finite H, the code $\mathrm{aHom}(G, H)$ is AlgEcon. The validity of such a statement depends not only on the class $\mathfrak{G}$ but also on the representation of the domain and the codomain.

▶ **Corollary 5.** *Let G be a finite abelian group and H an arbitrary finite group. Under suitable assumptions on the representation of G and H, the homomorphism code* $\mathrm{aHom}(G, H)$ *is AlgEcon.*

In other words, abelian groups are *universally AlgEcon.*

We need to clarify the "suitable representation." It suffices to have G in its primary decomposition and to have black-box access to H. These concepts, and other options for G, are discussed in Appendix A.

A *permutation representation of degree m* of a group G is a homomorphism $G \to S_m$, where the codomain is the symmetric group of degree m. We obtain efficient local list-decoding for the permutation representations of alternating groups under a rather generous restriction on the size of the permutation domain.

▶ **Theorem 6** (Main algorithmic result)**.** *Let $G = A_n$ be the alternating group of degree n and $H = S_m$ the symmetric group of degree m. Then* $\mathrm{aHom}(G, H)$ *is AlgEcon, assuming $m < 2^{n-1}/\sqrt{n}$.*

The limitations on the codomain arise from the severe technical difficulties encountered.

In contrast to all previous work, in the alternating case the minimum distance does not necessarily correspond to a subgroup of smallest index in the group G/N where N is the "irrelevant kernel," i. e., the intersection of the kernels of all $G \to H$ homomorphisms (see Sec. C). This necessitates the introduction of the *homomorphism extension* (HomExt) *problem*, a problem of interest in its own right, which remains the principal bottleneck for algorithmic progress.

By a $G \to H$ *partial map* we mean a function $\gamma\colon \mathrm{dom}(\gamma) \to H$ where $\mathrm{dom}(\gamma) \subseteq G$. The **homomorphism extension problem** HomExt asks whether a $G \to H$ partial map extends to a $G \to H$ homomorphism. The HomExt_λ problem asks this only for maps γ whose domain generates a subgroup of density $\mu(\langle \mathrm{dom}\,\gamma \rangle) > \lambda$ in G.

The HomExt_λ problem was solved by Wuu [21] in the special case required for Theorem 6.

1.2.3 Certificate list-decoding

To bypass the HomExt bottleneck, we introduce a new model we call **Certificate List-Decoding**. In this model the output is a short (poly$(1/\varepsilon)$-length) list of $G \to H$ partial maps that includes, for each affine homomorphism φ within ($\mathsf{mindist} - \varepsilon$) of the received word, a *certificate* of φ, i. e., a partial affine homomorphism that uniquely extends to φ.

We say that a homomorphism code is **economically certificate-list-decodable (CertEcon)** if such a list can be efficiently generated.

Note that, by definition, AlgEcon $\implies$ CertEcon $\implies$ CombEcon.

We say that a class $\mathfrak{G}$ of finite groups is **universally CertEcon** if for all $G \in \mathfrak{G}$ and arbitrary (finite or infinite) H, the code $\mathrm{aHom}(G, H)$ is CertEcon.

▶ **Theorem 7** (Main semi-algorithmic result). *Alternating groups are universally CertEcon.*

In fact we show that SRG groups are universally CertEcon.

By the *density* of a partial map γ we mean the density of the subgroup $\langle \mathrm{dom}(\gamma) \rangle$ in G. A λ-certificate list-decoder produces partial maps of density $\geq \lambda$. The $\textsc{HomExt}_\lambda$ problem asks to solve HomExt for partial maps of density $\geq \lambda$.

It is immediate that a λ-certificate list-decoder, combined with a $\textsc{HomExt}_\lambda$ solver, suffices for list-decoding $\mathrm{aHom}(G, H)$. This is the route we take to proving Theorem 6.

For a received word f and an affine homomorphism φ within distance $\mathsf{mindist} - \varepsilon$ of f, a **domain certificate** of φ is a subset S of the domain such that f restricted to S is a certificate of φ. Our semi-algorithmic results will actually produce lists of domain-certificates without requiring any access to the codomain.

1.2.4 Mean-list-decoding and domain extension

We define the (G, H)-*irrelevant kernel* N as the intersection of the kernels of all $G \to H$ homomorphisms. We show that if $\mathrm{aHom}(G/N, H)$ is CombEcon then so is $\mathrm{aHom}(G, H)$.

The corollaries include a CombEcon result for {arbitrary $\to$ abelian} homomorphism codes because of the known CombEcon result for {abelian $\to$ abelian} homomorphism codes [10]. More generally we have a CombEcon result for {arbitrary $\to$ nilpotent} homomorphism codes in view of the CombEcon result for {nilpotent $\to$ nilpotent} homomorphism codes [16, 9]). Analogous results hold for CertEcon and AlgEcon under suitable assumptions on access to the groups.

The main tool underlying these results is the notion of *mean-list-decoding*, where we study not the distance to one received word but the average distance to a family of received words. Our main result in this area establishes the equivalence of CombEcon for list-decoding and mean-list-decoding; and analogous results for CertEcon and AlgEcon.

We discuss these results in some detail in Appendix C. The mean-list-decoding technique was inspired by the concatenated code technique used in [16].

1.2.5 Hom versus aHom

The reader may ask, why we (and all prior work) consider affine homomorphisms rather than homomorphisms. The reason is that affine homomorphisms are the more natural objects in this context. First, this object is more homogeneous. For instance, for finite H, under random affine homomorphisms, the image of any element $g \in G$ is uniformly distributed over H. This uniformity also serves as an inductive tool: when extending the domain from a subgroup G_0 to a group G, the action of any homomorphism $\varphi \in \mathrm{Hom}(G, H)$ can be split into actions on the cosets of G_0 in G. Those actions are affine homomorphisms. On the other hand we also note that list-decoding $\mathrm{Hom}(G, H)$ and $\mathrm{aHom}(G, H)$ are essentially equivalent tasks.

▶ **Proposition 8** (Hom versus aHom). *Let G be a finite group, and H a group.*

(a) *[15, Prop. 2.5] If $|\mathrm{Hom}(G,H)| \geq 2$, then $\mathsf{mindist}(\mathrm{Hom}(G,H)) = \mathsf{mindist}(\mathrm{aHom}(G,H))$.*

(b) *For $X \in \{Comb, Cert, Alg\}$, if $\mathrm{Hom}(G,H)$ is $X Econ$ then $\mathrm{aHom}(G,H)$ is $X Econ$. For $X \in \{Cert, Alg\}$, this statement requires that nearly uniform random elements of G be available.*

▶ Remark. The length of the aHom list for distance $\mathsf{mindist} - \varepsilon$ is not greater than $\frac{1}{1-\mathsf{mindist}+\epsilon}$ times the length of the Hom list.

2 Notation and terminology

2.1 Group theoretic notation

Our general group theory reference is [19]. For the theory of permutation groups we refer to [11].

For finite sets $B \subseteq A$ where $A \neq \emptyset$, we write $\mu(B) = \mu_A(B) = |B|/|A|$ for the *density* of B in A. We use the notation $[n] = \{1, \ldots, n\}$.

For G and M groups, we write $M \leq G$ to indicate that M is a subgroup of G. For a group H and $T \subseteq H$, the *centralizer* $C_H(T)$ consists of those elements of H that commute with all elements of T. For $T \subseteq H$ we write $\langle T \rangle$ to denote the subgroup generated by T.

For a set Ω, the *symmetric group* $\mathrm{Sym}(\Omega)$ consists of all permutations of Ω. We write $S_n = \mathrm{Sym}([n])$. Permutation groups acting on Ω are subgroups of $\mathrm{Sym}(\Omega)$; their *degree* is $|\Omega|$. The *alternating group* $A_n \leq S_n$ consists of the even permutations. For $G \leq \mathrm{Sym}(\Omega)$ and $\Delta \subseteq \Omega$, the *pointwise stabilizer* $G_{(\Delta)}$ consists of those $\sigma \in G$ that fix Δ pointwise. The setwise stabilizer G_Δ is defined analogously.

2.2 Computational representation of groups

Our general reference to algorithmic group theory is [20].

Commonly used explicit representations include permutation groups, matrix groups, various representations of abelian groups such as the primary decomposition and the canonical form, etc. The latter are explained in Appendix A.1.

Black-box access is a general concept of oracle access to group operations. A black-box group is a finite group with (1) elements encoded by strings of uniform length, (2) black-box access, and (3) a given list of names of generators.

These concepts are explained in Appendix A.2.

3 Strategy

Let $f \in \mathrm{aHom}(G,H)$ be a received word and let $\mathcal{L}$ be the list of codewords within distance $(\mathsf{mindist} - \varepsilon)$ of f. The combinatorial problem is to find a bound of the form $|\mathcal{L}| \leq \mathrm{poly}(1/\varepsilon)$. First we use a sphere-packing argument to split $\mathcal{L}$ into a moderate number of *buckets* (more manageable subsets).

3.1 Notation: agreement, equalizer

The *agreement* $\mathrm{agr}(f,g)$ of two functions f, g in the code space H^G is the proportion of inputs on which f and g agree, i. e.,

$$\mathrm{agr}(f,g) = \frac{|\mathrm{Eq}(f,g)|}{|G|}, \tag{2}$$

where $\mathrm{Eq}(f,g) = \{x \in G \mid f(x) = g(x)\}$ is the *equalizer* (agreements set) of f and g. So, the distance between f and g is $1 - \mathrm{agr}(f,g)$. Following established notation, we write $\Lambda_{G,H}$ to denote the maximum agreement between pairs of distinct elements of $\mathrm{aHom}(G,H)$,

$$\Lambda_{G,H} = \max\{\mathrm{agr}(\varphi,\psi) \mid \varphi,\psi \in \mathrm{Hom}(G,H), \varphi \neq \psi\}, \tag{3}$$

or $\Lambda_{G,H} = 0$ if the only $G \to H$ homomorphism is the trivial one. So, the minimum distance of the code $\mathrm{Hom}(G,H)$ is $1 - \Lambda_{G,H}$. By Proposition 8(a), the minimum distance of $\mathrm{aHom}(G,H)$ is also $1 - \Lambda_{G,H}$. When G and H are clear from context, we write $\Lambda = \Lambda_{G,H}$.

For a homomorphism code $\mathcal{C}$ (either $\mathrm{Hom}(G,H)$ or $\mathrm{aHom}(G,H)$) and $\lambda > 0$, we write

$$\mathcal{L}(\mathcal{C}, f, \lambda) = \{\varphi \in \mathcal{C} \mid \mathrm{agr}(f,\varphi) \geq \lambda\}. \tag{4}$$

So the list $\mathcal{L}$ defined in the preamble of Section 3 is $\mathcal{L} = \mathcal{L}(\mathrm{aHom}(G,H), f, \Lambda + \epsilon)$.

3.2 A sphere packing argument

We shall use a sphere packing argument to split the list into more manageable parts. We begin with a strong negative correlation inequality.

▶ **Definition 9** (Strong negative correlation)**.** Let $\tau > 0$. Let $A_1, \ldots, A_k$ be events in a probability space. We say that $A_1, \ldots, A_k$ are **τ-strongly negatively correlated** if $\Pr(A_i \cap A_j) \leq \Pr(A_i)\Pr(A_j) - \tau$ for all $i \neq j$.

▶ **Lemma 10** (Strong negative correlation bound)**.** *Let $\tau > 0$. Let $A_1, \ldots, A_k$ be τ-strongly negatively correlated events in a probability space. Then $k \leq \frac{1}{4\tau} + 1$.*

Proof. For $1 \leq i \leq k$, let Z_i be the indicator random variable (characteristic function) of the event A_i; so $\mathbb{E}(Z_i) = \Pr(A_i)$ and $\mathrm{Var}(Z_i) = \Pr(A_i)(1 - \Pr(A_i)) \leq \frac{1}{4}$. For the covariances $(i \neq j)$ we have $\mathrm{Cov}(Z_i, Z_j) = \mathbb{E}[Z_i Z_j] - \mathbb{E}[Z_i]\,\mathbb{E}[Z_j] \leq -\tau$. So,

$$0 \leq \mathrm{Var}\left(\sum_i Z_i\right) = \sum_i \mathrm{Var}(Z_i) + \sum_{i \neq j} \mathrm{Cov}(Z_i, Z_j) \leq \frac{k}{4} - k(k-1)\tau. \tag{5}$$

Solving for k gives the bound as claimed. ◀

In our applications, P will be the uniform distribution μ over a finite set and we shall always have $\Pr(A_i) \geq \Lambda + \varepsilon$.

▶ **Lemma 11** (Sphere packing bound)**.** *Let G be a finite group, H a group, and $\varepsilon > 0$. Let $f\colon G \to H$ be a received word. Let $\Psi \subseteq \mathcal{L} = \mathcal{L}(\mathrm{aHom}(G,H), f, \Lambda + \varepsilon)$ be a subset of the list that is maximal under the constraint that $\mathrm{agr}(\psi_1, \psi_2) \leq \Lambda^2$ for all distinct $\psi_1, \psi_2 \in \Psi$. Then*

$$|\Psi| \leq \frac{1}{4(2\Lambda + \varepsilon)\varepsilon} + 1 \leq \frac{1}{4\varepsilon^2} + 1. \tag{6}$$

Proof. Observe that the sets $\mathrm{Eq}(\psi, f)$ for $\psi \in \Psi$ have density $\geq \Lambda + \varepsilon$ and they are $\varepsilon(2\Lambda + \varepsilon)$-strongly negatively correlated. Apply Lemma 10. ◀

▶ **Lemma 12.** *Let $\mathcal{L} = \mathcal{L}(\mathrm{aHom}(G,H), \Lambda + \varepsilon)$. If $|\mathcal{L}| \leq p(1/\Lambda, 1/\varepsilon)$ for some monotone function $p(.,.)$ then $|\mathcal{L}| \leq p(2/\varepsilon^2, 1/\varepsilon) + 1/(2\varepsilon^2)$. In particular, in the definition of CombEcon, we may replace the bound $\mathrm{poly}(1/\varepsilon)$ by $\mathrm{poly}(1/\Lambda, 1/\varepsilon)$ without changing the meaning.*

Proof. For $\Lambda > \varepsilon^2/2$, we are done. For $\Lambda \leq \varepsilon^2/2$ we have $|\mathcal{L}| \leq 1 + 1/(2\varepsilon^2)$ by Lemma 10 because the sets $\mathrm{Eq}(f, \varphi)$ for $\varphi \in \mathcal{L}$ are $(\varepsilon^2/2)$-strongly negatively correlated. ◀

For $\psi \in \Psi$, we define the bucket $\mathcal{L}_\psi$ by

$$\mathcal{L}_\psi = \{\varphi \in \mathcal{L} \mid \operatorname{agr}(\varphi, \psi) > \Lambda^2\} \tag{7}$$

The union of the buckets includes the list $\mathcal{L}$. Since the number of buckets is $|\Psi| \le 1 + 1/(4\varepsilon^2)$, we only need to bound the size of each bucket by $\operatorname{poly}(1/\varepsilon)$.

Our strategy to bound bucket size differs depending on the type of the domain G.

3.3 Bounding the list size for abelian groups

To prove that abelian groups are universally CombEcon, we prove that the codomain has a small number of abelian subgroups such that each homomorphism in the list $\mathcal{L}$ maps the domain G into one of those abelian subgroups. This reduces the problem to showing CombEcon for {abelian $\to$ abelian} homomorphism codes, which was done by Dinur, Grigorescu, Kopparty, and Sudan [10].

We work now with homomorphisms instead of affine homomorphisms; we will appeal to Proposition 8 to obtain affine results.

▶ **Theorem 13.** *Let G be a finite abelian group and H an arbitrary group. Let $f \in H^G$ be a received word. Then there exists a set $\mathcal{A}$ of finite abelian subgroups of the codomain H with $|\mathcal{A}| \le \dfrac{1}{4(2\Lambda+\varepsilon)\varepsilon^2} + \dfrac{1}{\varepsilon}$ such that for all $\varphi \in \mathcal{L} = \mathcal{L}(\operatorname{Hom}(G,H), f, \Lambda+\varepsilon)$, there is $M \in \mathcal{A}$ such that $\varphi(G) \le M$.*

We will find $\mathcal{A}$ by working separately on each bucket. We define, for each $\psi \in \Psi$, a set $\mathcal{A}_\psi$ of finite abelian subgroups of H such that

(i) for all $\varphi \in \mathcal{L}_\psi$, there is $M \in \mathcal{A}_\psi$ such that $\varphi(G) \le M$,

(ii) $|\mathcal{A}_\psi| \le 1/\varepsilon$.

It follows from (i) that we can set $\mathcal{A} = \bigcup_{\psi\in\Psi} \mathcal{A}_\psi$, so Theorem 13 follows from (ii) and the sphere packing bound (Lemma 11).

To define the set $\mathcal{A}_\psi$, we introduce the following concept. Let H be a group, $B \le H$ and $T \subseteq H$. The *abelian enlargement* of T by B is the group generated by T and the elements of B that commute with all elements of T, i. e.,

$$\operatorname{enl}_B(T) = \langle T, C_H(T) \cap B \rangle. \tag{8}$$

Note that if both $\langle T \rangle$ and B are finite abelian groups then so is $\operatorname{enl}_B(T)$; this is the only case in which we shall be interested. When $T = \{h\}$ is a singleton, we write $\operatorname{enl}_B(h)$ for $\operatorname{enl}_B(T)$.

Fix $\psi \in \Psi$. Let $\mathcal{A}_\psi$ be the set of all subgroups $M \le H$ that occur as $M = \operatorname{enl}_{\psi(G)}(\varphi(G))$ for some φ in the bucket $\mathcal{L}_\psi$. We shall show that every $M \in \mathcal{A}_\psi$ is equal to $\operatorname{enl}_{\psi(G)}(f(g))$ for at least an ε proportion of $g \in G$. The idea is that since φ and ψ have large agreement, most of $\varphi(G)$ is contained in $\psi(G)$. So even if we take a single random element $g \in G$, it is likely that the enlargement of $\varphi(g)$ by $\psi(G)$ already contains all of $\varphi(G)$. Specifically, we show the following.

▶ **Proposition 14.** *Let $\varphi, \psi \in \operatorname{Hom}(G,H)$ and $g \in G$ such that $\langle g, \operatorname{Eq}(\psi,\varphi)\rangle = G$. Then $\varphi(G) \le \operatorname{enl}_{\psi(G)}(\varphi(G)) = \operatorname{enl}_{\psi(G)}(\varphi(g))$.*

And, since f and φ have high agreement, it is likely that $\varphi(g) = f(g)$. We elaborate on this strategy in Section 4.

3.4 Bucket estimation for alternating groups

Subgroups of polynomial index in alternating groups are well understood; they are described by a result known as the Jordan–Liebeck Theorem (JLT), see [11, Theorem 5.2A].

▶ **Theorem 15** (Jordan–Liebeck). *Let $n \geq 10$ and let r be an integer with $1 \leq r < n/2$. Suppose $K \leq A_n$ has index $|A_n : K| < \binom{n}{r}$. Then for some $\Delta \subseteq [n]$ with $|\Delta| < r$ we have $(A_n)_{(\Delta)} \leq K \leq (A_n)_\Delta$.*

Here $(A_n)_{(\Delta)}$ denotes the pointwise stabilizer of Δ in A_n and $(A_n)_\Delta$ the setwise stabilizer of Δ.

We use JLT multiple times in this section.

Ignoring the trivial case $\Lambda = 0$, it is easy to show that for $G = A_n$ with $n \geq 5$ we have $\Lambda \geq 1/\binom{n}{2}$. It then follows from JLT that for $n \geq 10$ we have $\Lambda = 1/\binom{n}{s}$ for $s \in \{1, 2\}$. In the light of Lemma 12 it suffices to find a $\mathrm{poly}(n, 1/\varepsilon)$ bound on the size of each bucket.

3.4.1 Nonconstructive proof

Let $\mathcal{K}$ denote the set of all subgroups that are the pointwise stabilizer of $2s$ points:

$$\mathcal{K} = \{(A_n)_{(\Delta)} \mid \Delta \subseteq [n], |\Delta| = 2s\} \tag{9}$$

where $s \in \{1, 2\}$ and $\Lambda = 1/\binom{n}{s}$ (see above).

We shall refer to the elements of $\mathcal{K}$ as *label subgroups*. We have $|\mathcal{K}| = \binom{n}{2s}$. By JLT for $n \geq 11$, every subgroup of A_n of index $< \binom{n}{2s+1}$ contains a member of $\mathcal{K}$. It is not difficult to see that the depth of any $K \in \mathcal{K}$ in A_n is 5 if $s = 2$ (cf. [1]) and 2 if $s = 1$. (All we need is that this depth is bounded, which is obvious.)

All homomorphisms φ in the bucket $\mathcal{L}_\psi$ have agreement $> \Lambda^2$ with one representative homomorphism ψ; so φ and ψ agree on a subgroup of index $< 1/\Lambda^2 \leq \binom{n}{2s+1}$ (for $n \geq 40$) and therefore, by JLT, they agree on some label subgroup. So we can split each bucket $\mathcal{L}_\psi$ further into $\binom{n}{2s}$ *sub-buckets* $\mathcal{L}_{\psi,K}$, where the homomorphisms in $\mathcal{L}_{\psi,K}$ agree with ψ on K.

Bound on the size of sub-buckets. To bound the size of a sub-bucket $\mathcal{L}_{\psi,K}$, we describe a process for choosing a random homomorphism in the sub-bucket. For a positive integer d, we choose d random elements of G. If there is a unique homomorphism φ that agrees with f on the d random inputs, and agrees with ψ on K, we choose this homomorphism.

Now we combine the following two straightforward observations.

▶ **Observation 16.** *Let $K \leq G$ be a subgroup, $\psi \in \mathrm{Hom}(G, H)$ a homomorphism, d a nonnegative integer, and $g_1, \ldots, g_d \in G$. If $\mu(\langle K, g_1, \ldots, g_d\rangle) > \Lambda$, then there is at most one homomorphism $\varphi \in \mathrm{Hom}(G, H)$ such that $K \leq \mathrm{Eq}(\psi, \varphi)$ and $g_1, \ldots, g_d \in \mathrm{Eq}(f, \varphi)$.*

▶ **Observation 17.** *Let $0 \leq \lambda < 1$. Let G be a finite group, $K \leq G$ a subgroup, and $S \subseteq G$ a subset. Suppose $\mu(S) > \lambda$. Let $\varepsilon = \mu(S) - \lambda$ and $d = \mathrm{depth}_G(K)$. Then,*

$$\Pr_{g_1,\ldots,g_d \in G}[g_1, \ldots, g_d \in S \text{ and } \mu(\langle K, g_1, \ldots, g_d\rangle) > \lambda] \geq \varepsilon^d. \tag{10}$$

It follows that if $d \geq \mathrm{depth}_{A_n} K$ ($K \in \mathcal{K}$) then each homomorphism in the sub-bucket $\mathcal{L}_{\psi,K}$ gets chosen with probability at least ε^d and therefore $|\mathcal{L}_{\psi,K}| \leq 1/\varepsilon^d$. By the foregoing, we may choose $d = 2s + 1$, so $|\mathcal{L}_{\psi,K}| \leq 1/\varepsilon^{2s+1}$.

Combining our bounds on the number of buckets, the number of sub-buckets per bucket, and size of each sub-bucket, we conclude that $|\mathcal{L}| = O(\varepsilon^{-2s-3}\Lambda^{-2})$ and therefore $|\mathcal{L}| = O(\varepsilon^{-2s-7}) = O(\varepsilon^{-11})$ by Lemma 12.

This concludes our first proof that the alternating groups are universally CombEcon. The proof is non-constructive because it relies on the sphere packing argument.

3.4.2 Constructive proof

Our second strategy for list decoding {alternating → arbitrary} exploits a property of the alternating groups which we call *shallow random generation* (SRG).

A group G is SRG if, roughly, a small number of random elements of G are extremely likely to generate a low-depth subgroup.

▶ **Definition 18** (Shallow random generation)**.** Let $k, d \in \mathbb{N}$. We say that a finite group G is (k,d)**-shallow generating** if

$$\Pr_{g_1,\ldots,g_k \in G}[\mathrm{depth}(\langle g_1,\ldots,g_k\rangle) > d] < (\Lambda^*_G)^k, \tag{11}$$

where $\Lambda^*_G = \min_H\{\Lambda_{G,H} \mid \Lambda_{G,H} \neq 0\}$.

A class $\mathfrak{G}$ of finite groups has **shallow random generation** ($\mathfrak{G}$ is SRG) if there exist $k, d \in \mathbb{N}$ such that all $G \in \mathfrak{G}$ are (k,d)-shallow generating.

Alternating groups are an example of a class of SRG groups.

▶ **Theorem 19.** *The class of alternating groups is SRG. Specifically, for sufficiently large n, the group A_n is $(2,5)$-shallow generating.*

The proof uses the following result that says that two random elements of an alternating group are extremely likely to act as an alternating or symmetric group on a large subset of the permutation domain [6].

▶ **Theorem 20** (Babai)**.** *Let π, σ be a pair of independent uniform random elements from S_n. For $0 \le t \le n/3$, let $E(n,t)$ denote the following event: The subgroup $K = \langle \pi, \sigma\rangle$ acts as S_r or A_r on r elements of the permutation domain for some $r \ge n - t$. Then,*

$$\Pr(E(n,t)) = 1 - \binom{n}{t+1}^{-1} + O\left(\binom{n}{t+2}^{-1}\right). \tag{12}$$

The constant implied by the big-O notation is absolute.

To prove Theorem 19 we use Theorem 20 with $t = 4$, noting that $\Lambda^*_{A_n} = 1/\binom{n}{2}$ and $\mathrm{depth}_{A_n}(A_{n-4}) = 5$.

From Observations 16 and 17 one can infer that SRG classes of groups are CombEcon. This view allows us not only to combinatorially list-decode {SRG → arbitrary}, but also to certificate list-decode; certificates are given by f restricted to a small number of random elements of G.

▶ **Definition 21.** A domain certificate $S \subseteq G$ is a *domain-Λ-certificate* if $\mu(\langle S\rangle) > \Lambda$.

▶ **Theorem 22** (SRG implies CertEcon, via domain certificates)**.** *Let $k \in \mathbb{N}$ and $c > 0$. Let G be a (k,d)-shallow generating group and H a group. Let $f\colon G \to H$ and $\varepsilon > 0$. Let Υ be a list of $\left\lceil \frac{1}{\varepsilon^{k+d}} \ln\left(\frac{4}{\varepsilon^{k+d}}\right)\right\rceil$ independently chosen subsets of G, each of size $k+d$. Then, with probability at least $3/4$, Υ is a domain-Λ-certificate-list of $\mathcal{L}(\mathrm{aHom}(G,H), f, \Lambda + \varepsilon)$.*

▶ Remark. From Theorem 19 with $k = 2$, $d = 5$ we infer that for $\{A_n \to \text{arbitrary}\}$, the length of the list is $\widetilde{O}(\varepsilon^{-7})$.

We say that a class $\mathfrak{G}$ is λ-*CertEcon* if it is CertEcon with the additional requirement that for all certificates γ in our certificate list, we must have $\mu(\langle \operatorname{dom} \gamma \rangle) > \lambda$.

▶ **Theorem 23.** *If $\mathfrak{G}$ is an SRG class of groups, then $\mathfrak{G}$ is universally Λ-CertEcon.*

Here we assume that the domain G is given as a black-box group. In order to obtain domain-certificates, no access to the codomain H is needed. To obtain actual certificates ($G \to H$ partial functions), we only need black-box access to H.

3.5 Cert + HomExt = Alg

A Λ-certificate list-decoder and a $\textsc{HomExt}_\Lambda$ solver combine to an algorithmic list decoder. Wuu [21] solved the homomorphism extension problem in the following case.

▶ **Theorem 24** (Wuu). *Assume $m < 2^{n-1}/\sqrt{n}$ and $\lambda = 1/\operatorname{poly}(n)$. Then the $\textsc{HomExt}_\lambda(A_n, S_m)$ search problem can be solved in $\operatorname{poly}(n, m)$ time.*

Combining the previous two theorems, we obtain our main algorithmic result.

▶ **Theorem 25** (Main algorithmic result). *$\operatorname{aHom}(A_n, S_m)$ is AlgEcon, assuming $m < 2^{n-1}/\sqrt{n}$.*

4 Homomorphism Codes with finite abelian domain and arbitrary codomain

In this section we describe the details of the proof that finite abelian groups are universally combinatorially and algorithmically economically list-decodable. The key technical result is Theorem 13, which says that there are a small number of abelian subgroups of the codomain such that every homomorphism in the list maps into one of these subgroups.

In Section 4.1, we state characterizations of $\Lambda_{G,H}$ when G is abelian. In Section 4.2 we state facts about abelian enlargements (see definition in Section 3.3). Using this tool, in Section 4.3 we prove Theorem 13 (the key result mentioned in the previous paragraph) and infer that abelian groups are universally CombEcon. In Section 4.4 we adapt the algorithm of [10, 16], to give an algorithm to locally list-decode these codes.

We remark that these codes usually cannot be list-decoded beyond radius $1 - (\Lambda_{G,H} + \varepsilon)$ (see Remark at the end of Section 1.2.1).

4.1 $\Lambda_{G,H}$ when G is abelian

The following characterization of $\Lambda_{G,H}$ for G abelian is clear.

▶ **Fact 26.** Let G be a finite abelian group and H a group. The following are equivalent for any prime p.

(a) $\Lambda_{G,H} = 1/p$.

(b) p is the smallest prime number such that p divides $|G|$ and H has an element of order p.

(c) p is the smallest prime number dividing $|G : N|$, where N is the (G, H)-irrelevant kernel.

If no such p exists in (b) or (c), then $|\operatorname{Hom}(G, H)| = 1$ and $\Lambda_{G,H} = 0$.

Guo [15, Theorem 1.1] gave a characterization of $\Lambda_{G,H}$ when G and H are finite groups with G solvable or H nilpotent.

4.2 Abelian enlargements

Throughout this section, let G be a finite abelian group, and H a group (finite or infinite). We prove facts about abelian enlargements, which were defined in Section 3.3.

▶ **Lemma 27.** *Let $B \leq H$ a finite abelian subgroup, and $T \subseteq H$ a subset such that $\langle T \rangle$ is a finite abelian group. For $U \subseteq \mathrm{enl}_B(T)$, we have that $\mathrm{enl}_B(T) = \mathrm{enl}_B(T \cup U)$.*

Proof. First, we show that $\mathrm{enl}_B(T) \leq \mathrm{enl}_B(T \cup U)$. Since $\mathrm{enl}_B(T)$ is abelian, we have that

$$C_H(T) \cap B \leq \mathrm{enl}_B(T) \leq C_H(\mathrm{enl}_B(T)) \leq C_H(U). \tag{13}$$

So,

$$C_H(T) \cap B \leq C_H(T) \cap C_H(U) \cap B = C_H(T \cup U) \cap B \leq \mathrm{enl}_B(T \cup U). \tag{14}$$

Since also $T \subseteq \mathrm{enl}_B(T \cup U)$, we have that $\mathrm{enl}_B(T) \leq \mathrm{enl}_B(T \cup U)$.

Next, we show that $\mathrm{enl}_B(T \cup U) \leq \mathrm{enl}_B(T)$. We have that $T \subseteq \mathrm{enl}_B(T)$, that $U \subseteq \mathrm{enl}_B(T)$, and $C_H(T \cup U) \cap B \leq C_H(T) \cap B \leq \mathrm{enl}_B(T)$. So, $\mathrm{enl}_B(T \cup U) = \langle T \cup U, C_H(T \cup U) \cap B \rangle \leq \mathrm{enl}_B(T)$. ◀

▶ **Proposition 28.** *Let $\varphi, \psi \in \mathrm{Hom}(G, H)$ and $A \subseteq G$ such that $\langle A, \mathrm{Eq}(\psi, \varphi), \ker \varphi \rangle = G$. Then $\mathrm{enl}_{\psi(G)}(\varphi(A)) = \mathrm{enl}_{\psi(G)}(\varphi(G))$.*

Proof. Since G is finite abelian, so are $\varphi(G)$ and $\psi(G)$. Let $B = \psi(G)$. Let $T = \varphi(A)$. Let $U = \varphi(\mathrm{Eq}(\psi, \varphi))$. Since $T, U \subseteq \varphi(G)$, and $\varphi(G)$ is abelian, $U \leq C_H(T)$. And, since $U = \psi(\mathrm{Eq}(\psi, \varphi))$, we have that $U \leq \psi(T) = B$. Thus, $U \leq C_H(T) \cap B \leq \mathrm{enl}_B(T)$.

Also, $\langle T \cup U \rangle = \langle T, U, 1 \rangle = \langle \varphi(A), \varphi(\mathrm{Eq}(\psi, \varphi)), \varphi(\ker \varphi) \rangle = \varphi(\langle A, \mathrm{Eq}(\psi, \varphi), \ker \varphi \rangle) = \varphi(G)$.

Therefore, by Lemma 27,

$$\mathrm{enl}_{\psi(G)}(\varphi(A)) = \mathrm{enl}_B(T) = \mathrm{enl}_B(T \cup U) = \mathrm{enl}_B(\langle T \cup U \rangle) = \mathrm{enl}_{\psi(G)}(\varphi(G)). \tag{15}$$

◀

▶ **Corollary 29.** *Let φ, ψ, and A be as above. Then $\varphi(G) \leq \mathrm{enl}_{\psi(G)}(\varphi(A))$.*

Proposition 14 is a special case of Proposition 28.

4.3 Combinatorial list-decodability, finite abelian to arbitrary

In this section, we establish that finite abelian groups are universally CombEcon.

Throughout this section, let G be a finite abelian group, and H an arbitrary group (finite or infinite). Let $f \colon G \to H$ be a received word. Let $\varepsilon > 0$. Let $\mathcal{L} = \mathcal{L}(\mathrm{Hom}(G, H), f, \Lambda + \varepsilon)$ be the list (note that in this section we deal with the code of homomorphisms, rather than affine homomorphisms; however, we can convert between the two; see Section 1.2.5). The list $\mathcal{L}$ is divided into buckets $\mathcal{L}_\psi$ for $\psi \in \Psi$, where Ψ is as in Lemma 11.

We will see that there is a small set of abelian subgroups $M \leq H$ such that every $\varphi \in \mathcal{L}$ has its image in some M. Dinur, Grigorescu, Kopparty, and Sudan [10] proved that $\mathrm{aHom}(G, H)$ is CombEcon (and in fact, AlgEcon) for all finite abelian groups G and H. Theorem 13, combined with the DGKS result, lets us conclude that $\mathrm{Hom}(G, H)$ (and thus $\mathrm{aHom}(G, H)$) is CombEcon.

▶ **Corollary 30.** *Finite abelian groups are universally CombEcon. Specifically, let C be a constant such that $|\mathcal{L}(\mathrm{aHom}(G,H),f,\Lambda+\varepsilon)| \le (\frac{1}{\varepsilon})^C$ for G,H finite abelian groups. Then $|\mathcal{L}(\mathrm{aHom}(G,H),f,\Lambda+\varepsilon)| \le O((\frac{1}{\varepsilon})^{C+4})$ for G a finite abelian group and H an arbitrary group.*

By [16, 9], the constant C currently stands at ≈ 105.

Proof of Corollary 30. Let $\mathcal{A}$ be the collection of subgroups of H guaranteed by Theorem 13. Then, $\mathcal{L} \subseteq \bigcup_{M\in\mathcal{A}} \mathcal{L}(\mathrm{Hom}(G,M),f,\Lambda+\varepsilon)$ (on the right hand side we let f be redefined arbitrarily at points in its domain that do not map to M). So,

$$|\mathcal{L}| \le \sum_{M\in\mathcal{A}} \ell(\mathrm{Hom}(G,M),\Lambda+\varepsilon) \le \left(\frac{1}{4(2\Lambda+\varepsilon)\varepsilon^2}+\frac{1}{\varepsilon}\right)\left(\frac{1}{\varepsilon}\right)^C. \tag{16}$$

We then apply the Remark after Proposition 8. ◀

In the remainder of this subsection, we prove Theorem 13.

Let Ψ be as in Lemma 11. Recall our strategy from Section 3.2 of dividing the list $\mathcal{L}$ into buckets $\mathcal{L}_\psi$ for $\psi\in\Psi$. We will prove the following.

▶ **Lemma 31.** *Let $\psi\in\Psi$. There is a set $\mathcal{A}_\psi$ of finite abelian subgroups of H with $|\mathcal{A}_\psi| \le \frac{1}{\varepsilon}$ such that for all $\varphi\in\mathcal{L}_\psi$, there is $M\in\mathcal{A}_\psi$ for which $\varphi(G)\le M$.*

From this, Theorem 13 follows by taking $\mathcal{A} = \bigcup_{\psi\in\Psi}\mathcal{A}_\psi$.

Proof of Lemma 31. Let $\mathcal{A}_\psi = \{\mathrm{enl}_{\psi(G)}(\varphi(G)) \mid \varphi\in\mathcal{L}_\psi\}$. Then $\mathcal{A}_\psi$ is a set of finite abelian subgroups of H. And, for all $\varphi\in\mathcal{L}_\psi$, we have that $\varphi(G)\le \mathrm{enl}_{\psi(G)}(\varphi(G))\in\mathcal{A}_\psi$.

Let a be a uniform random element of G. For each $M\in\mathcal{A}_\psi$, let E_M be the event that $\mathrm{enl}_{\psi(G)}(f(a)) = M$. We will show that $\Pr[E_M]\ge\varepsilon$. Since the events E_M for $M\in\mathcal{A}_\psi$ are pairwise disjoint, this will imply that $|\mathcal{A}_\psi|\le\frac{1}{\varepsilon}$.

Consider any $M\in\mathcal{A}_\psi$. There exists $\varphi\in\mathcal{L}_\psi$ such that $\mathrm{enl}_{\psi(G)}(\varphi(G)) = M$. Let N be the (G,H)-irrelevant kernel. Since $N\le\mathrm{Eq}(\varphi,\psi)$, we have that $|G:\mathrm{Eq}(\psi,\varphi)|$ divides $|G:N|$, whose smallest prime factor is $\frac{1}{\Lambda}$ by Fact 26.

If $a\in\mathrm{Eq}(f,\varphi)\smallsetminus\mathrm{Eq}(\psi,\varphi)$, then $\mu(\langle a,\mathrm{Eq}(\psi,\varphi)\rangle) \ge \frac{1}{\Lambda}\mu(\mathrm{Eq}(\psi,\varphi)) > \frac{1}{\Lambda}\Lambda^2 = \Lambda$, so $\langle a,\mathrm{Eq}(\psi,\varphi)\rangle = G$. In this case, by Proposition 14,

$$\mathrm{enl}_{\psi(G)}(f(a)) = \mathrm{enl}_{\psi(G)}(\varphi(a)) = \mathrm{enl}_{\psi(G)}(\varphi(G)) = M. \tag{17}$$

Therefore,

$$\Pr[E_M] \ge \Pr[a\in\mathrm{Eq}(f,\varphi)\smallsetminus\mathrm{Eq}(\psi,\varphi)] \ge \mathrm{agr}(f,\varphi)-\mathrm{agr}(\psi,\varphi)\ge\varepsilon. \tag{18}$$

We conclude that $|\mathcal{A}_\psi|\le\frac{1}{\varepsilon}$. ◀

4.4 Algorithm

▶ **Definition 32.** A **primary decomposition** of a finite abelian group G is a representation as a direct product of cyclic groups of prime-power order.

For G a finite abelian group and H an arbitrary group, we can locally list-decode $\mathrm{aHom}(G,H)$. Based on our CombEcon bound for this class of pairs of groups, we adapt the algorithm of Dinur, Grigorescu, Kopparty, and Sudan from [10, Sec. 5]. Thus, such codes are AlgEcon. Like [10], we assume that G is given explicitly by an primary decomposition.

▶ **Theorem 33.** *Let $\mathfrak{D}$ be the class of pairs (G, H) where G is a finite abelian group given explicitly by an primary decomposition, and H is a group with black-box access. Then there is an algorithm to locally list-decode $\mathfrak{D}$ in time* $\mathrm{poly}(\log|G| \cdot \frac{1}{\varepsilon})$.

Here we assume the unit-cost model of naming elements of H (cf. Def. 34).

Details of our adaptation of the algorithm of Dinur et al. [10] are given in Appendix D.

References

1 László Babai. On the length of subgroup chains in the symmetric group. *Communications in Algebra*, 14(9):1729–1736, 1986. doi:10.1080/00927878608823393.

2 László Babai. Local expansion of vertex-transitive graphs and random generation in finite groups. In *23rd STOC*, pages 164–174. ACM, 1991. doi:10.1145/103418.103440.

3 László Babai, Robert Beals, and Ákos Seress. Polynomial-time theory of matrix groups. In *41st STOC*, pages 55–64. ACM, 2009. doi:10.1145/1536414.1536425.

4 László Babai, Timothy J. F. Black, and Angela Wuu. List-decoding homomorphism codes with arbitrary codomains. *arXiv*, 2018. (full version of this paper). arXiv:1806.02969.

5 László Babai and Endre Szemerédi. On the complexity of matrix group problems I. In *25th FOCS*, pages 229–240. IEEE Computer Soc., 1984. doi:10.1109/SFCS.1984.715919.

6 László Babai. The probability of generating the symmetric group. *J. Combinat. Theory, Series A*, 52(1):148–153, 1989. doi:10.1016/0097-3165(89)90068-X.

7 Robert Beals, Charles R. Leedham-Green, Alice C. Niemeyer, Cheryl E. Praeger, and Ákos Seress. Constructive recognition of finite alternating and symmetric groups acting as matrix groups on their natural permutation modules. *J. Algebra*, 292(1):4–46, 2005. doi:10.1016/j.jalgebra.2005.01.035.

8 Abhishek Bhowmick and Shachar Lovett. The list decoding radius of Reed-Muller codes over small fields. In *47th STOC*, pages 277–285. ACM, 2015. doi:10.1145/2746539.2746543.

9 Timothy Black, Alan Guo, Madhu Sudan, and Angela Wuu. List decoding nilpotent groups. In preparation, 2018.

10 Irit Dinur, Elena Grigorescu, Swastik Kopparty, and Madhu Sudan. Decodability of group homomorphisms beyond the Johnson bound. In *40th STOC*, pages 275–284. ACM, 2008. doi:10.1145/1374376.1374418.

11 John D. Dixon and Brian Mortimer. *Permutation Groups*. Graduate Texts in Math. Springer New York, 1996.

12 Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *21st STOC*, pages 25–32. ACM, 1989. doi:10.1145/73007.73010.

13 Parikshit Gopalan, Adam R. Klivans, and David Zuckerman. List-decoding Reed-Muller codes over small fields. In *40th STOC*, pages 265–274. ACM, 2008. doi:10.1145/1374376.1374417.

14 Elena Grigorescu, Swastik Kopparty, and Madhu Sudan. Local decoding and testing for homomorphisms. In *APPROX-RANDOM*, volume 4110 of *Lecture Notes in Computer Science*, pages 375–385. Springer, 2006. doi:10.1007/11830924_35.

15 Alan Guo. Group homomorphisms as error correcting codes. *Electronic J. Combinatorics*, 22(1):P1.4, 2015.

16 Alan Guo and Madhu Sudan. List decoding group homomorphisms between supersolvable groups. In *APPROX/RANDOM*, volume 28 of *LIPIcs*, pages 737–747. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2014. doi:10.4230/LIPIcs.APPROX-RANDOM.2014.737.

17 William Cary Huffman and Vera Pless. *Fundamentals of Error-Correcting Codes*. Cambridge Univ. Press, 2003.

18 I. Martin Isaacs. *Finite Group Theory*. Graduate studies in mathematics. Amer. Math. Soc., 2008.
19 Derek J. S. Robinson. *A Course in the Theory of Groups*. Springer, 2nd edition, 1995.
20 Ákos Seress. *Permutation Group Algorithms*. Cambridge Tracts in Math. Cambridge Univ. Press, 2003.
21 Angela Wuu. Homomorphism extension. *arXiv*, 2018. arXiv:1802.08656.

A Computational representation of groups

A.1 Finite abelian groups

For finite abelian groups, the representation used by prior authors is the *primary decomposition*, i. e., the representation as the direct product of cyclic groups of prime power order.

The *canonical form* of finite abelian groups is the representation as the direct product of cyclic groups of orders $n_1, \ldots, n_k$ where $n_i \mid n_{i+1}$. Note that any *abelian presentation* in terms of generators and relations can be converted, in polynomial time, to the canonical form using the Smith normal form of integer matrices.

The canonical representation alone will not suffice for the algorithms in prior work; we need be able to factor the n_i in order to convert this to primary representation. This can be done, for instance, if *a superset of the prime divisors of the order of the finite abelian group G is available.*

A.2 Black-box access, black-box groups

Our most general model of access to a group is *black-box access*. In this model, a "universe" U is a collection of potential names of group elements. Not all elements of the universe encode group elements and a group element may have multiple names.

▶ **Definition 34.** We say that we have **black-box access** to a group G if the following holds. There is a set U of "names" and a surjection $r\colon U \to G \cup *$ where $*$ is a special symbol. For $u \in r^{-1}(U)$ we say that u is a name of $r(u) \in G$. Given the names of two group elements, an oracle gives a name of their product/quotient, and recognizes the names of the identity element. We assume a tape that can hold an element of U in each cell, accessible at unit cost ("unit-cost model").

Black-box access does not assume we know the name of any element of G. Nevertheless, such access may be sufficient in the case of the codomain; in this case we assume the received word consists of names of elements of the codomain. "Black-box groups" (introduced in [5]) require in addition that a list of generators be know, and that the names be words of uniform length over a fixed finite alphabet.

▶ **Definition 35.** We say that a *finite group* G is given as a **black-box group** if the following hold.

(a) The universe U is Σ^n where Σ is a fixed finite alphabet. We call n the *encoding length* of the group elements.
(b) G is given black-box access over the universe U.
(c) The names of a list of generators of G are given.

It follows in particular that $|G| \leq |\Sigma|^n$.

An important consequence of representation as a black-box group is the availability of nearly uniform random elements. We say that we sample elements of a finite set Ω ϵ-nearly uniformly if the probability of each element to be selected is between $(1 \pm \epsilon)/|\Omega|$.

▶ **Theorem 36** ([2]). *Given a finite group G as a black-box group, one can generate names of independent, nearly uniformly distributed elements of G; the per element cost is polynomial in the encoding length of the elements and $|\log(\epsilon)|$ where ϵ is the near-uniformity parameter.*

In our applications, setting $\epsilon = 1/2$ will suffice.

B Upper bound on the list-decoding radius

We have shown that the class of homomorphism codes with alternating domain and arbitrary codomain have list-decoding radius greater than $1 - (\Lambda + \varepsilon)$ for all $\varepsilon > 0$.

On the other hand, we now show that in many cases, a blowup of the list size occurs at radius $1 - \Lambda$, demonstrating that the list-decoding radius is at most $1 - \Lambda$ in these cases.

The number of homomorphisms within a closed ball of radius $1 - \Lambda$ of a received word will be exponential in $\log|G|$ and $\log|H|$. We note that $|H| \geq |G|$ unless $\Lambda = 0$.

▶ **Proposition 37.** *For any n, and $\lambda \in \{1/n, 1/\binom{n}{2}\}$, there exists a finite group H_n such that $\Lambda_{A_n,H_n} = \lambda$ and*

$$\ell(\mathrm{Hom}(A_n, H_n), \Lambda) = 2^{\Omega(n)} \geq 2^{\Omega\left(\sqrt[3]{\log|H|}\right)}. \tag{19}$$

Moreover, for any fixed $n \geq 10$, and any integer M, there is a finite group H such that

$$\ell(\mathrm{Hom}(A_n, H), \Lambda) \geq M. \tag{20}$$

Proof. We use the same construction for both parts. To prove the first claim, let $k = n$. To prove the second claim, let $k \geq \log_2 M$.

Suppose $\lambda = 1/n$. Let $H_n = A_{n+1}^k$, the direct product of k copies of A_{n+1}. Then $\Lambda_{A_n,H_n} = 1/n$. Let $f\colon A_n \to H_n$ by $f(g) = (g, \ldots, g)$, the diagonal identity map, where A_n is embedded in A_{n+1}. For nonempty $S \subseteq [n]$ and $j \in [n]$, let $h = h(S, j) = (h_1, \ldots, h_k) \in H_n$, where h_i is the transposition $(j, n+1)$ if $i \in S$ and 1 otherwise. For each such h, let $\varphi_h \in \mathrm{Hom}(A_n, H_n)$ be given by $\varphi_h(g) = h^{-1} f(g) h$. Each φ_h has agreement $\mathrm{agr}(\varphi_h, f) = 1/n = \Lambda$ with f. There are $n(2^k - 1)$ such h, so $\ell(\mathrm{Hom}(A_n, H_n), \Lambda) \geq n(2^k - 1)$.

Suppose $\lambda = 1/\binom{n}{2}$. Let $H_n = A_n^k$. Then, $\Lambda_{A_n,H_n} = 1/\binom{n}{2}$. Let $f\colon A_n \to H_n$ by $f(g) = (g, \ldots, g)$, the diagonal identity map. For nonempty $S \subseteq [n]$ and $\tau \in S_n$ is a transposition, let $h = h_{S,\tau} = (h_1, \ldots, h_k) \in A_n^k$, where $h_i = \tau$ if $i \in S$ and 1 otherwise. For each such h, let $\varphi_h \in \mathrm{Hom}(A_n, H_n)$ be given by $\varphi_h(g) = h^{-1} f(g) h$. Each such φ_h has agreement $\mathrm{agr}(\varphi_h, f) = 1/\binom{n}{2}$. There are $\binom{n}{2}(2^k - 1)$ such h, so $\ell(\mathrm{Hom}(A_n, H_n), \Lambda) \geq \binom{n}{2}(2^k - 1)$. ◀

We remark that $\ell(\mathrm{Hom}(A_n, H), \Lambda_{A_n,H})$ is not bounded as a function of n for a wide variety of classes of H.

C Mean-list-decoding, irrelevant kernel, and domain relaxation

In this section we discuss *mean-list-decoding*, a tool for extending our economical list-decoding results to wider classes of domain groups. The metric used in mean-list-decoding is not distance to a single received word, but rather the average distance to a family of received words. Although apparently more general than list-decoding, mean-list-decoding is actually

equivalent to list-decoding (in the CombEcon, CertEcon, and AlgEcon sense). An important implication is that we can extend our list-decoding results to a wider class of domain groups. Specifically, define the (G, H)-*irrelevant kernel* to be the intersection of the kernels of all $G \to H$ homomorphisms. We show (Theorem 40) that for $\mathrm{aHom}(G, H)$ to be economically list-decodable (CombEcon, CertEcon, or AlgEcon), it suffices to show that $\mathrm{aHom}(G/N, H)$ is. As an example, the {abelian $\to$ abelian} list-decoding results automatically extend to {arbitrary $\to$ abelian} results.

For a family $\mathcal{F} = \{f_i \mid i \in I\}$ of received words and a word $f \in H^G$, we define the **average agreement** $\mathrm{agr}(w, \mathcal{F})$ of w and $\mathcal{F}$ by

$$\mathrm{agr}(w, \mathcal{F}) = \frac{1}{|\mathcal{F}|} \sum_{i \in I} \mathrm{agr}(w, f_i). \tag{21}$$

For a homomorphism code $\mathcal{C}$ (either $\mathrm{Hom}(G, H)$ or $\mathrm{aHom}(G, H)$), the **mean-list** $\mathcal{L}$ is the set of codewords whose agreement with $\mathcal{F}$ is at least a specified quantity ρ; i. e.,

$$\mathcal{L} = \mathcal{L}(\mathcal{C}, \mathcal{F}, \rho) = \{w \in \mathcal{C} \mid \mathrm{agr}(w, \mathcal{F}) \geq \rho\}. \tag{22}$$

We define **CombEconM**, **CertEconM**, and **AlgEconM** by replacing the received word f with a family of received words $\mathcal{F}$ and replacing "the list" with "the mean-list" in the definitions of CombEcon, CertEcon, and AlgEcon, respectively. However, the -M concepts turn out to be equivalent to the non-M, concepts.

▶ **Theorem 38.** *For a class $\mathcal{C}$ of homomorphism codes, $\mathcal{C}$ is CombEconM if and only if it is CombEcon.*

Under suitable access assumptions, analogous results hold for CertEcon and AlgEcon.

Theorem 38 follows from the following observation.

▶ **Lemma 39.** *For all homomorphism codes $\mathcal{C}$ (either $\mathrm{Hom}(G, H)$ or $\mathrm{aHom}(G, H)$), for all families $\mathcal{F}$ of received words, for all $\rho, \delta > 0$,*

$$|\mathcal{L}(\mathcal{C}, \mathcal{F}, \rho)| \leq \frac{1}{\delta} \max_{f \in \mathcal{F}} |\mathcal{L}(\mathcal{C}, f, \rho + \delta)|. \tag{23}$$

For groups G and H, and $N \trianglelefteq G$ a subgroup of the (G, H)-irrelevant kernel, we note that any homomorphism (or affine homomorphism) $G \to H$ is the composition of a homomorphism (or affine homomorphism) $G/N \to H$ with the projection map $G \to G/N$. Thus, $\Lambda_{G,H} = \Lambda_{G/N,H}$.

▶ **Theorem 40.** *Let G, H be groups and $N \trianglelefteq G$ a subgroup of the (G, H)-irrelevant kernel. If $\mathrm{aHom}(G/N, H)$ is CombEcon, then $\mathrm{aHom}(G, H)$ is CombEcon.*

▶ Remark. Under suitable access assumptions, analogous results hold for CertEcon and AlgEcon. These assumptions involve (a) uniform random generation of elements of N, and (b) oracle access to a transversal, i. e., an injection $G/N \to G$ that assigns a representative element to each coset.

Proof of Theorem 40. Let $\Lambda = \Lambda_{G,H} = \Lambda_{G/N,H}$.

We prove the theorem in the combinatorial setting. Fix a set S of coset representatives of N in G. To the function $f \colon G \to H$ we associate the family $\mathcal{F} = \{f_n \colon G/N \to H \mid n \in N\}$ where $f_n(sN) = f(sn)$ for all $n \in N$ and $s \in S$. Then, $\mathrm{agr}(\varphi \circ \pi, f) = \mathrm{agr}(\varphi, \mathcal{F})$ for all $\varphi \in \mathrm{aHom}(G/N, H)$, where $\pi \colon G \to G/N$ is the projection map. If we identify $\varphi \in \mathrm{aHom}(G/N, H)$ with $\varphi \circ \pi \in \mathrm{aHom}(G, H)$, then, for any $\epsilon > 0$, we have

$$\mathcal{L}(\mathrm{aHom}(G, H), f, \Lambda + \epsilon) = \mathcal{L}(\mathrm{aHom}(G/N, H), \mathcal{F}, \Lambda + \epsilon). \tag{24}$$

Now Theorem 40 follows by an application of Theorem 38. ◀

D Adaptation of the DGKS algorithm to abelian → arbitrary

In Section 4.4 we mentioned that to prove the {abelian→arbitrary} AlgEcon result, we combine our CombEcon result for this class of codes with an adaptation of the {abelian → arbitrary} algorithm from [10]. Here we indicate how our version differs from that algorithm.

First, [10] reduces to the case where $H = \mathbb{Z}_{p^r}$. We do not make such a reduction. We let p be the prime such that $\Lambda = \frac{1}{p}$. Every mention of $\mathbb{Z}_{p^r}$ should be replaced by H. As in their algorithm, we take $G = G_1, \ldots, G_k$, with each $G_i = \mathbb{Z}_{p_i^{r_i}}$. We order the G_i such that $p_1 = p$. For them, the only important coordinates are the ones where $p_i = p$, but for our purposes, instances of $\mathbb{Z}_{p^{r_i}}$ should be replaced with $\mathbb{Z}_{p_i^{r_i}}$.

In the algorithm EXTEND of [10], the statement "If $c_1 - c_2$ is not divisible by p" should be replaced with "If $c_1 - c_2$ is not divisible by p_i, and if $f(y_1, c_1, s)$ and $f(y_2, c_2, s)$ commute with each other and with $\varphi(e_1), \ldots, \varphi(e_{i-1})$." Here e_j denotes a generator of G_j. The system of equations that follows should be solved under the assumption that the order of a divides $p_i^{r_i}$.

We note that when solving the system of equations in EXTEND, we are working in an abelian subgroup of H. Actually, even this does not matter; we can solve the given system of equations without assuming the elements of H commute.

In the algorithm as stated, we assume that the value $\Lambda_{G,H}$ is known. This assumption can actually be discarded.

Optimal Deterministic Extractors for Generalized Santha-Vazirani Sources

Salman Beigi
Institute for Research in Fundamental Sciences, Tehran, Iran

Andrej Bogdanov
Chinese University of Hong Kong

Omid Etesami
Institute for Research in Fundamental Sciences, Tehran, Iran

Siyao Guo
Northeastern University, Boston, USA

Abstract

Let $\mathcal{F}$ be a finite alphabet and $\mathcal{D}$ be a finite set of distributions over $\mathcal{F}$. A Generalized Santha-Vazirani (GSV) source of type $(\mathcal{F}, \mathcal{D})$, introduced by Beigi, Etesami and Gohari (ICALP 2015, SICOMP 2017), is a random sequence $(F_1, \ldots, F_n)$ in $\mathcal{F}^n$, where F_i is a sample from some distribution $d \in \mathcal{D}$ whose choice may depend on $F_1, \ldots, F_{i-1}$.

We show that all GSV source types $(\mathcal{F}, \mathcal{D})$ fall into one of three categories: (1) non-extractable; (2) extractable with error $n^{-\Theta(1)}$; (3) extractable with error $2^{-\Omega(n)}$.

We provide essentially randomness-optimal extraction algorithms for extractable sources. Our algorithm for category (2) sources extracts one bit with error ε from $n = \mathrm{poly}(1/\varepsilon)$ samples in time linear in n. Our algorithm for category (3) sources extracts m bits with error ε from $n = O(m + \log 1/\varepsilon)$ samples in time $\min\{O(m2^m \cdot n), n^{O(|\mathcal{F}|)}\}$.

We also give algorithms for classifying a GSV source type $(\mathcal{F}, \mathcal{D})$: Membership in category (1) can be decided in NP, while membership in category (3) is polynomial-time decidable.

2012 ACM Subject Classification Theory of computation → Expander graphs and randomness extractors, Mathematics of computing → Probability and statistics, Mathematics of computing → Information theory

Keywords and phrases feasibility of randomness extraction, extractor lower bounds, martingales

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.30

Acknowledgements Andrej Bogdanov's work was supported by HK RGC GRF grants CUHK 14208215 and CUHK 14238716. Siyao Guo's work is supported by NSF grants CNS1314722 and CNS-1413964. Part of this work was done while Omid Etesami and Siyao Guo were visiting the Chinese University of Hong Kong, and while Andrej Bogdanov and Siyao Guo were visiting the Simons Institute for the Theory of Computing at UC Berkeley. We would like to thank the reviewers for constructive comments.

1 Introduction

A randomness extractor is an algorithm that converts a weak source of randomness into almost uniform independent random bits. One of the first classes of distributions that were considered in the context of randomness extraction are Santha-Vazirani (SV) sources [16], also called unpredictable-bit sources. An SV source is a sequence of random bits such that every bit in the sequence has entropy bounded away from zero, even when conditioned on

any possible sequence of previous bits. As already pointed out in [16], deterministic (seedless) extraction of even a single almost unbiased bit from SV sources is impossible, although these sources have entropy that grows linearly with their length.[1]

The motivating application for randomness extraction is the simulation of randomized algorithms when only a weak random source is available. For stand-alone algorithms seeded extractors are sufficient to accomplish this simulation. The simulation runs the extractor over all possible seeds and observes the resulting outcomes of the algorithm. In distributed and cryptographic settings, however, multiple executions may be infeasible. In contrast, deterministic extractors of sufficiently high quality are adequate for all these applications.

Even for stand-alone algorithms, simulation using a seeded extractor incurs costs in running time and statistical error. A seed of length ℓ entails 2^ℓ executions of the algorithm and can at best guarantee a statistical error of $2^{-\ell/2}$ [15]. In cryptographic settings, errors are typically inverse-polynomial in the complexity of the adversary, so a simulation of this type may be infeasible. Dodis et al. [9] prove that many cryptographic tasks including encryption, zero-knowledge, secret-sharing, and two-party computation are impossible from arbitrary high-entropy sources including SV sources.

1.1 Generalized Santha-Vazirani sources

In this work we consider deterministic extraction for a natural generalization of Santha-Vazirani sources which was introduced by Beigi, Etesami, and Gohari [2, 3]. A *generalized Santha-Vazirani (GSV) source* is specified by a pair $(\mathcal{F}, \mathcal{D})$, where $\mathcal{F}$ is a finite set of *faces* and $\mathcal{D}$ is a finite set of *dice*, each of which is a probability distribution on $\mathcal{F}$. (We will assume that each face is assigned positive probability by at least one die.) A distribution $(F_1, \ldots, F_n)$, where the F_is are $\mathcal{F}$-valued correlated random variables, is admissible by the source if it is generated by the following type of *strategy*: For each $1 \leq i \leq n$, a die $d \in \mathcal{D}$ is chosen as a function of $F_1, \ldots, F_{i-1}$ and F_i is sampled according to the distribution d. The adversary's choice of the die d may be probabilistic as well (without changing the extractability from the source family).

The case $|\mathcal{F}| = 2$ recovers the definition of SV sources: The dice are two-sided coins, one biased towards heads and the other one towards tails. In this special case the condition $|\mathcal{D}| = 2$ can be imposed without loss of generality by convexity.

▶ **Definition 1.** We call a GSV source $(\mathcal{F}, \mathcal{D})$ *extractable* with error ε from n samples if there exists a function $\mathrm{Ext}\colon \mathcal{F}^n \to \{-1, 1\}$ such that for every distribution $(F_1, \ldots, F_n)$ in the source, $|\mathbb{E}[\mathrm{Ext}(F_1, \ldots, F_n)]| \leq \varepsilon$. We call a source *extractable* if for every error $\varepsilon > 0$ there exists a sample size n for which the source is extractable with these parameters.

The work [3] showed that randomness extraction from a GSV source is possible assuming the following condition:

▶ **Definition 2.** A GSV source $(\mathcal{F}, \mathcal{D})$ satisfies the *Nonzero Kernel Positive Variance (*NK^+*)* condition if there exists a function $\psi\colon \mathcal{F} \to [-1, 1]$ such that $\mathbb{E}_d[\psi(F)] = 0$ and $\mathrm{Var}_d[\psi(F)] > 0$ for every die $d \in \mathcal{D}$.

Here, $\mathbb{E}_d$ and Var_d denote expectation and variance with respect to the distribution of die d. On the other hand, they showed that extractability from such sources necessitates the following *Nonzero Kernel* (NK) condition:

There exists a nonzero $\psi\colon \mathcal{F} \to [-1, 1]$ such that $\mathbb{E}_d[\psi(F)] = 0$ for every die $d \in \mathcal{D}$.

[1] With respect to seeded extraction, a constant seed length is sufficient for all SV sources [17].

In particular, when all faces of all dice have positive probability (an assumption called "nondegeneracy" in [3]), the (NK^+) and (NK) conditions coincide, providing a characterization of extractability for this class of sources. Their extractor requires $\Theta(1/\varepsilon^3)$ samples to achieve error ε.

There are, however, simple examples of GSV sources ((E1) and (E2) below) that satisfy (NK) but not (NK^+). The work [3] does not address the extractability of such sources.

The existence of extractors for GSV sources does not appear to easily follow from counting arguments, as is the case of other types of sources for which extraction is known to be possible in principle and the focus is on efficient constructions, such as affine sources [6, 12], polynomial sources [11, 10] and independent blocks [5, 7].

1.2 Our Contributions

Our first contribution is a complete characterization of extractability from GSV sources. To motivate our result, we first observe that the (NK) condition is, in general, insufficient for extractability. Consider, for instance the two-diced, three-faced GSV source described by the distributions (i.e., probability mass functions) $d_1 = (0, 0, 1)$ and $d_2 = (\frac{1}{2}, \frac{1}{2}, 0)$. This source satisfies (NK) with the witness $\psi = (-1, 1, 0)$, but is clearly not extractable as the distribution in which d_1 is repeatedly tossed contains no entropy.

The following GSV source is a slightly more interesting example:

$$d_1 = (\tfrac{1}{2}, \tfrac{1}{2}, 0, 0) \quad d_2 = (0, 0, \tfrac{1}{3}, \tfrac{2}{3}) \quad d_3 = (0, 0, \tfrac{2}{3}, \tfrac{1}{3}). \tag{E1}$$

This source also satisfies the (NK) condition (with $\psi = (-1, 1, 0, 0)$). However, it is not extractable because it contains a "hidden" SV source (over two faces): If die d_1 is discarded and the first two faces are removed, dice d_2 and d_3 now fail the (NK) condition.

These two examples suggest the following method for coming up with non-extractable GSV sources: Start with any source that fails (NK), extend the dice with more faces of zero probability, and add any number of dice that assign positive probability to the new faces. To describe such sources, we introduce the following natural strengthening of (NK):

▶ **Definition 3.** A GSV source $(\mathcal{F}, \mathcal{D})$ satisfies the *Hereditary Nonzero Kernel (HNK)* condition if for every subset $\mathcal{D}' \subseteq \mathcal{D}$ there exists a nonzero function $\psi : \mathcal{F}' \to [-1, 1]$ such that $\mathbb{E}_d[\psi(F)] = 0$ for all $d \in \mathcal{D}'$, where $\mathcal{F}' = \mathcal{F}'(\mathcal{D}')$ is the set of faces to which at least one die in $\mathcal{D}'$ assigns nonzero probability.

Clearly (HNK) is a necessary condition for extractability, because if $(\mathcal{F}, \mathcal{D})$ fails (HNK) then $(\mathcal{F}', \mathcal{D}')$ fails (NK). Our first theorem shows that (HNK) is also sufficient. Moreover, it gives a universal upper bound on the number of samples:

▶ **Theorem 4.** *The following conditions are equivalent for a GSV source* $(\mathcal{F}, \mathcal{D})$*:*

1. $(\mathcal{F}, \mathcal{D})$ *satisfies HNK.*
2. $(\mathcal{F}, \mathcal{D})$ *is extractable.*
3. *For every* ε*,* $(\mathcal{F}, \mathcal{D})$ *is extractable with error* ε *from* $n = \mathrm{poly}(1/\varepsilon)$ *samples in time linear in* n*.*

In the course of proving Theorem 4 we introduce the analytic *Mean Variance Ratio (MVR)* condition that turns out to be equivalent to HNK (Proposition 12). We show that a quantitative variant of the MVR condition determines the best-possible quality of extraction, up to a quadratic gap, even for GSV sources that are not extractable to within arbitrary small error (Propositions 7 and 10).

The work [3] proved statement 3 of Theorem 4 for the subclass of NK^+ sources. Theorem 5 shows that NK^+ are in fact extractable from a logarithmic number of samples, and they are the only sources for which this degree of efficiency is possible.

▶ **Theorem 5.** *The following conditions are equivalent for a GSV source $(\mathcal{F}, \mathcal{D})$:*

1. *$(\mathcal{F}, \mathcal{D})$ satisfies NK^+.*
2. *For every ε, $(\mathcal{F}, \mathcal{D})$ is extractable with error ε from $o(1/\varepsilon^2)$ samples.*
3. *For every ε and m, $(\mathcal{F}, \mathcal{D})$ is extractable with error[2] ε and output length m from $n = O(\log(1/\varepsilon) + m)$ samples in time $\min\{O(m2^m \cdot n), n^{O(|\mathcal{F}|)}\}$.*

The sample complexity of the extractor in part 3 of Theorem 5 is optimal up to the leading constant: $\Omega(m)$ samples are necessary by entropy considerations, and $\Omega(1/\varepsilon)$ samples are necessary for non-trivial sources[3] by granularity considerations.

Condition NK^+ is strictly stronger than condition HNK. For example, the source

$$d_1 = (\tfrac{1}{2}, \tfrac{1}{2}, 0, 0), \quad d_2 = (\tfrac{1}{4}, \tfrac{1}{12}, \tfrac{1}{3}, \tfrac{1}{3}), \quad d_3 = (\tfrac{1}{12}, \tfrac{1}{4}, \tfrac{1}{3}, \tfrac{1}{3}). \tag{E2}$$

satisfies HNK but not NK^+.

Taken together, Theorems 4 and 5 completely classify non-trivial GSV sources into three categories: (1) non-extractable, (2) extractable with error $n^{-\Theta(1)}$, and (3) extractable with error $2^{-\Omega(n)}$, where n is the number of samples. This rules out the existence of GSV sources of other error rates like $1/\log n$ or $2^{-\sqrt{n}}$.

Moreover, sources can be classified algorithmically: Condition HNK can be decided by a coNP algorithm, while NK^+ is polynomial-time decidable (see Proposition 18).

Figure 1 indicates the relations between the different conditions for extractability of GSV sources uncovered in this work. The left and right columns describe equivalent formulations of randomness-efficient and general extractability, respectively. The middle column contains the different types of GSV sources in increasing generality from top to bottom. The HNK condition is shown equivalent to all formulations of general extractability, while the NK^+ condition [3] is shown equivalent to randomness-efficient extractability.

1.3 Proof Techniques

Feasibility of extraction. The extractor of [3] outputs the sign of $Z_T = \psi(F_1) + \cdots + \psi(F_T)$ at the earliest time T when $|Z_T|$ exceeds some pre-specified threshold M. Here, ψ is the witness for condition (NK^+), which ensures that $\mathbb{E}[\psi(F)]$ is always zero and $\mathrm{Var}[\psi(F)]$ is always positive. Therefore (Z_t) is a martingale with growing variance, and the analysis of [3] shows that the process terminates by time $n = O(1/\varepsilon^3)$ except with probability $\varepsilon/2$ when M is chosen as $\Theta(1/\varepsilon)$. Moreover, Z_T must take value in the range $(-(M+1), -M] \cup [M, M+1)$, so by the optional stopping time theorem, the bias of Z_T is $\varepsilon/2$ when $M = \Theta(1/\varepsilon)$.

In case only the weaker (HNK) condition holds, $\mathrm{Var}[\psi(F)]$ could be zero for some dice and the value of Z_t may remain constant throughout the process. On the other hand, (HNK) provides not one but many witnesses ψ, one for every subset of the dice. Proposition 12 shows how all these witnesses can be combined into a single $\phi\colon \mathcal{F} \to [-1, 1]$ that has positive variance with respect to all the dice, but may have nonzero expectation. By a careful

[2] The error of an extractor that outputs multiple bits is the statistical (total variation) distance between its output distribution and the uniform distribution.

[3] The exception consists of GSV sources that admit an event of probability exactly half after throwing only one die, for which errorless extraction is possible.

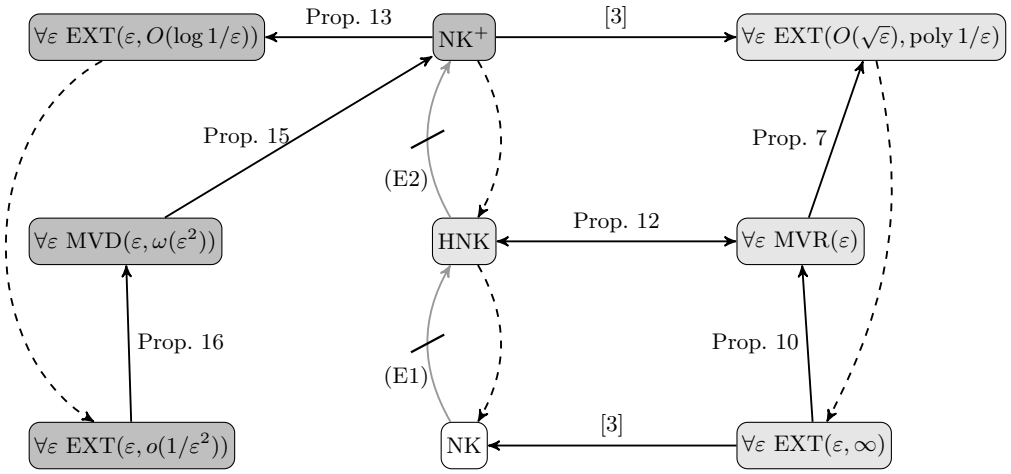


Figure 1 A map of our results. Straight arrows are implications (the dashed ones are immediate) and lighter struck-out arrows are separations given by Examples (E1) and (E2). EXT(ε, n) postulates extractability with error ε from n samples. Lightly and darkly shaded boxes represent equivalent conditions for extractability and randomness-efficient extractability, respectively. Definitions 2, 3, 6, and 14 specify the NK$^+$, HNK, MVR, and MVD conditions, respectively.

implementation of this strategy, it is ensured that the ratio $|\mathbb{E}_d[\phi(F)]|/\mathrm{Var}_d[\phi(F)]$ can be made smaller than any pre-specified $\varepsilon > 0$. This is our Mean Variance Ratio (MVR) condition. Moreover, $\mathrm{Var}_d[\phi(F)]$ can be lower bounded by ε^C for some constant C that depends only on the GSV source.

To prove Theorem 4 we apply the extractor of [3] to the function ϕ. As ϕ may be biased with respect to some dice, (Z_t) may no longer be a martingale, rendering the optional stopping time theorem inapplicable. In Proposition 7 we demonstrate that the conclusion of the [3] analysis still applies in our context. Intuitively, the (MVR) condition should imply that the variance of Z_t grows, and does so at a faster rate than the magnitude of its expectation. Therefore the stopping time should still be finite, and the component of extraction error incurred by $|\mathbb{E}[Z_T]|$ should be small. Owing to dependencies between the various steps, a rigorous implementation of these ideas requires substantial care.

Quality and quantity of extracted bits. For GSV sources that satisfy (NK$^+$) the extractor of [3] inherently requires $\Omega(1/\varepsilon)$ samples: On the one hand, to ensure termination with high probability the boundary threshold M can be at most n, but on the other hand Z_T may fall anywhere in the range $(-(M+1), -M] \cup [M, M+1)$, thereby incurring an error of $\varepsilon = \Omega(1/M)$.[4] To improve the sample complexity, our bit extractor in Theorem 5 applies the update rule

$$Z_{t+1} = Z_t + \frac{\psi(F_t)}{2} \cdot (1 - |Z_t|)$$

and outputs the sign of Z_n for $n = O(\log 1/\varepsilon)$. Under (NK$^+$) the sequence (Z_t) is still a martingale, but now the range of Z_t is restricted to the open interval $(-1, 1)$. On average, the deviation of the step size $Z_{t+1} - Z_t$ conditioned on Z_t is smaller the closer Z_t is to one

[4] A tempting alternative is to simply output the sign of Z_n after looking at some predetermined number of samples. However, this "extractor" incurs error $\Omega(1)$ for any non-trivial GSV source.

of the boundary points $\{-1, 1\}$. We show that the logarithm of $1/(1 - |Z_t|)$ grows by a constant on average in every step and apply Azuma's inequality to conclude that Z_n is within $2^{-\Omega(n)}$ of 1 or -1 with probability $1 - 2^{-\Omega(n)}$. This ensures the bias of the output is inverse exponential in the number of samples.

We give some informal intuition about the fast convergence of the martingale. Since its range is bounded to $[-1, 1]$ and it is "unstable" at any point far from the boundaries, by Doob's theorem it must converge to either -1 or 1. The extractor's decision about its output should not be influenced heavily by the last few symbols of the source. Otherwise, the extractor can be biased by an adversarial choice of distribution. (In particular, a random function is typically a poor extractor since the last bits have very high influence.) A "sufficient statistic" for the extractor's future strategy at any point is the bias of its output given the symbols read so far. This probability is not a well-defined quantity since the source is selected adversarially. The (NK$^+$) condition enables a rigorous (and optimal) implementation of this strategy.

To extract multiple bits, the state $\mathbf{Z}_t$ of the above process is extended to encode a probability distribution over $\{0, 1\}^m$. Initially $\mathbf{Z}_0$ is the uniform distribution. The distance measure $1 - |Z_t|$ is replaced by a carefully chosen quantity $\mathbf{D}_t \in \mathbb{R}^{2^m}$ which ensures that $\mathbf{Z}_t$ is a probability distribution that rapidly concentrates on a single entry in $\{0, 1\}^m$, which is the output of the extractor. Since $(\mathbf{Z}_t)$ is a multi-dimensional martingale, the output must be statistically close to uniform. The straightforward method of keeping track of the corresponding 2^m martingales requires exponential space, but we show a way to maintain a succinct representation of them.

Lower bounds. Beigi, Etesami, and Gohari [3] proved that if a source fails the (NK) condition, then it is not extractable. There are two proofs of this fact in [3], an analytic one (in Section 2.2) and a combinatorial one (in Appendix B). Here we refine the ideas of the analytic proof. In Proposition 10 we prove a quantitatively precise refinement of this statement: As we shall show, the (NK) condition fails if for all ψ there exists a die d for which $|\mathbb{E}_d[\psi(F)]|/\mathrm{Var}_d[\psi(F)] = \Omega(1)$. Now if $|\mathbb{E}_d[\psi(F)]|/\mathrm{Var}_d[\psi(F)] \geq \varepsilon$, then the extraction error must be at least $\Omega(\varepsilon)$. We conclude that extractability implies the (MVR) condition, which together with a compactness argument (see Proposition 12) gives (HNK), proving the "only if" direction of Theorem 4.

We prove the "only if" direction of Theorem 5 in Section B. We introduce the mean-variance divergence (MVD) condition, which postulates that $|\mathbb{E}_d[\psi(F)]| < \varepsilon(\mathrm{Var}_d[\psi(F)] - \delta)$ for all dice. In Proposition 16 we show that if MVD fails then extraction with error ε requires $\Omega(1/\delta)$ samples. In Proposition 15 we use linear-algebraic duality to show that if (NK$^+$) fails then so does (MVD) with $\delta = O(\varepsilon^2)$, thereby completing the proof of Theorem 5.

1.4 Other related work

The question of extractability from GSV sources has found applications in tampering attacks in cryptography [14, 13] and in publicly verifiable randomness via cryptocurrencies [4].

The GSV sources also capture the block sources of [8]. The latter (ℓ, b)-sources correspond to the special case of $\mathcal{F} = \{0, 1\}^\ell$ and $\mathcal{D}$ containing all flat distributions over $\mathcal{F}$ having min-entropy at least b. Thus our theorems refine the impossibility of deterministic extraction from block sources.

2 A characterization of extractable GSV sources

In this Section we prove Theorem 4. The following analytic condition plays a central role in the proof:

▶ **Definition 6.** A GSV source $(\mathcal{F}, \mathcal{D})$ satisfies the *Mean-Variance Ratio condition* with parameter $\varepsilon > 0$ (MVR(ε)) if there exists a function $\psi : \mathcal{F} \to [-1, 1]$ such that for every die $d \in \mathcal{D}$ of a GSV source $(\mathcal{F}, \mathcal{D})$,

$$\left|\mathbb{E}_d[\psi(F)]\right| < \epsilon \mathrm{Var}_d[\psi(F)]. \tag{MVR}$$

Proposition 7 in Section 2.1 shows that if a GSV source satisfies MVR(ε) then it is extractable with error $O(\sqrt{\varepsilon})$ from poly$(1/\varepsilon)$ samples. On the other hand, Proposition 10 in Section 2.2 shows that any GSV source that is extractable with error less than $\varepsilon/10$ (from any number of samples) satisfies MVR(ε). Thus the smallest ε for which MVR(ε) holds measures the best-possible quality of extraction of a GSV source to within a square.

In the case when MVR(ε) holds for all $\varepsilon > 0$, the source is extractable. Proposition 12 shows that if this is the case then HNK must hold. HNK, in turn, implies a slightly stronger form of "MVR(ε) for all ε". Together with Proposition 7 this establishes the extractability of HNK sources from ε^{-C} samples, where C is a constant that depends only on the source.

2.1 Feasibility of extraction

▶ **Proposition 7.** *If GSV source $(\mathcal{F}, \mathcal{D})$ satisfies* MVR(ε)*, then it is extractable from n samples with error at most $3\sqrt{\varepsilon} + 4/\varepsilon v n + O(\varepsilon)$, where v is the minimum of* $\mathrm{Var}_d[\psi(F)]$ *over all $d \in \mathcal{D}$.*

The extractor outputs the sign of $\psi(F_1) + \cdots + \psi(F_T)$ if $T \le n$, where F_i is the i-th output of the GSV source sequence, and T is the first time when the magnitude of this expression exceeds the value $M = 1/\sqrt{\varepsilon}$. The output can be arbitrary if $T > n$.

The sequence $X_t = \psi(F_t)$ is a special case of an (ε, v)-*approximate martingale*, namely a sequence (X_t), $|X_t| \le 1$ such that

$$|\mathbb{E}[X_t \mid X_{<t}]| < \varepsilon \cdot \mathrm{Var}[X_t \mid X_{<t}] \qquad \text{and} \qquad \mathrm{Var}[X_t \mid X_{<t}] \ge v$$

for all t and $X_{<t} = (X_1, \ldots, X_{t-1})$. The key result is the following lemma which bounds the deviation time of an approximate martingale and its bias at the deviation time.

▶ **Lemma 8.** *Let $Z_t = X_1 + \cdots + X_t$ and T be the first time when $|Z_T| \ge M$ for an (ε, v)-approximate martingale (X_t). Then $|\mathbb{E}[Z_T]| \le \varepsilon(M+1)^2$ and $\Pr[T > t] \le 2(M+1)^2/vt$, assuming $\varepsilon \le 1/8(M+1)$.*

Proof of Lemma 8. We modify the sequence so that $X_t = 0$ for all $t > T$. The variables of interest T and Z_T do not change, while the approximate martingale assumptions imply

$$|\mathbb{E}[X_t \mid X_{<t}]| \le \varepsilon \cdot \mathrm{Var}[X_t \mid X_{<t}] \qquad \text{and} \qquad \mathrm{Var}[X_t \mid X_{<t}, T > t] \ge v.$$

The advantage of this modification is that $|Z_t|$ is now upper bounded by $M+1$ for all t. The upper bounds on $\Pr[T > t]$ and $|\mathbb{E}[Z_T]|$ will both follow from this variance lower bound:

▶ **Claim 9.** *For all t,* $\mathrm{Var}[Z_t] \ge \frac{1}{2}\sum_{i=1}^{t} \mathbb{E}[\mathrm{Var}[X_i | X_{<i}]]$.

Proof. By the law of total variance, for every t,

$$\mathrm{Var}[Z_t] = \mathrm{Var}\big[\mathbb{E}[Z_t|X_{<t}]\big] + \mathbb{E}\big[\mathrm{Var}[Z_t|X_{<t}]\big].$$

Furthermore,

$$\begin{aligned}\mathrm{Var}\big[\mathbb{E}[Z_t|X_{<t}]\big] &= \mathrm{Var}\big[Z_{t-1} + \mathbb{E}[X_t|X_{<t}]\big] \\ &= \mathrm{Var}[Z_{t-1}] + \mathrm{Var}\big[\mathbb{E}[X_t|X_{<t}]\big] + 2\mathrm{Cov}\big(Z_{t-1}, \mathbb{E}[X_t|X_{<t}]\big).\end{aligned}$$

We can lower bound the covariances by

$$\begin{aligned}\mathrm{Cov}\big(Z_{t-1}, \mathbb{E}[X_t|X_{<t}]\big) &= \mathbb{E}\big[\,(Z_{t-1} - \mathbb{E}[Z_{t-1}]) \cdot \mathbb{E}[X_i|X_{<t}]\,\big] \\ &\geq -\mathbb{E}\big[\,|Z_{t-1} - \mathbb{E}[Z_{t-1}]| \cdot |\mathbb{E}[X_t|X_{<t}]|\,\big] \\ &\geq -\mathbb{E}\big[\,(2M+2) \cdot \epsilon \mathrm{Var}[X_t|X_{<t}]\,\big] \\ &\geq -\frac{1}{4}\mathbb{E}\big[\mathrm{Var}[X_t|X_{<t}]\big],\end{aligned}$$

where the penultimate inequality follows from the boundedness of Z_{t-1} and the last one follows from the assumption $\varepsilon \leq 1/8(M+1)$.

Combining the above (in)equalities and using the nonnegativity of $\mathrm{Var}[\mathbb{E}[X_t|X_{<t}]]$,

$$\begin{aligned}\mathrm{Var}[Z_t] &\geq \mathbb{E}\big[\mathrm{Var}[Z_t|X_{<t}]\big] + \mathrm{Var}[Z_{t-1}] - \frac{1}{2}\mathbb{E}\big[\mathrm{Var}[X_t|X_{<t}]\big] \\ &= \mathrm{Var}[Z_{t-1}] + \frac{1}{2}\mathbb{E}\big[\mathrm{Var}[X_t|X_{<t}]\big].\end{aligned}$$

The claim now follows by induction on t. ◀

We can now upper bound $|\mathbb{E}[Z_T]|$ by the maximum of $\mathbb{E}[Z_t]$ over all t, which is at most

$$|\mathbb{E}[Z_t]| \leq \sum_{i=1}^{t} \mathbb{E}\big[|\mathbb{E}[X_i|X_{<i}]|\big] \leq \sum_{i=1}^{t} \mathbb{E}\big[\varepsilon \mathrm{Var}[X_i|X_{<i}]\big] \leq 2\varepsilon \mathrm{Var}[Z_t] \leq 2\varepsilon(M+1)^2. \tag{1}$$

The second inequality follows from the approximate martingale assumption. The third one follows from Claim 9. The fourth one follows from the boundedness of Z_t.

It remains to upper bound $\Pr[T > t]$. By Claim 9, the law of conditional expectations, and the approximate martingale assumption,

$$\begin{aligned}\mathrm{Var}[Z_t] &\geq \sum_{i=1}^{t} \mathbb{E}\big[\mathrm{Var}[X_i|X_{<i}]\big] \\ &= \frac{1}{2}\sum_{i=1}^{t} \Pr[T > i] \cdot \mathbb{E}\big[\mathrm{Var}[X_i|X_{<i}] \,\big|\, T > i\big] \\ &\geq \frac{1}{2}\sum_{i=1}^{t} \Pr[T > t] \cdot v \\ &= \frac{tv}{2} \cdot \Pr[T > t].\end{aligned}$$

The desired lower bound follows from the boundedness of Z_t. ◀

Proof of Proposition 7. Let $X_t = \psi(F_t)$, $Z_t = X_1 + \cdots + X_t$ and T be the first time when $|Z_T| \geq M$. Conditioned on $T \leq n$, the output Ext of the extractor is identically distributed to $\mathrm{sign}(Z_T)$. Therefore by the law of conditional expectations,

$$|\mathbb{E}[\mathrm{Ext}] - \mathbb{E}[\mathrm{sign}(Z_T)]| = |\mathbb{E}[Ext \mid T > n] - \mathbb{E}[Z_T \mid T > n]| \cdot \Pr[T > n] \leq 2\Pr[T > n].$$

By the boundedness of the X_ts,

$$|Z_T - M \cdot \mathrm{sign}(Z_T)| \leq 1.$$

By the triangle inequality and Lemma 8,

$$\begin{aligned}|\mathbb{E}[\mathrm{Ext}]| &\leq |\mathbb{E}[\mathrm{sign}(Z_T)]| + 2\Pr[T > n] \\ &\leq \frac{|\mathbb{E}[Z_T]| + 1}{M} + 2\Pr[T > n] \\ &\leq \frac{\varepsilon(M+1)^2}{M} + \frac{1}{M} + \frac{4(M+1)^2}{vn},\end{aligned}$$

assuming $\varepsilon \leq 1/8(M+1)$. When $M = 1/\sqrt{\varepsilon}$ the assumption is satisfied for every $\varepsilon \leq 1/4$ and the desired bound follows. When $\varepsilon > 1/4$ the claimed bias is larger than one and there is nothing to prove. ◀

2.2 Impossibility of extraction

▶ **Proposition 10.** *Let ε be a sufficiently small constant. Assume* $\mathrm{MVR}(\varepsilon)$ *fails for a source $(\mathcal{F}, \mathcal{D})$. Then $(\mathcal{F}, \mathcal{D})$ is not extractable with error better than $\varepsilon/10$ from any number of samples.*

Proof of Proposition 10. Assuming $\mathrm{MVR}(\varepsilon)$ fails we prove the following claim:

▶ **Claim 11.** *For every n, every extractor* $\mathrm{Ext}\colon \mathcal{F}^n \to \{0,1\}$*, and every $0 \leq \alpha \leq 1$, if $\mathbb{E}_{A_-}[\mathrm{Ext}] \geq \alpha$ for every strategy A_-, then there exists a strategy A_+ for which $\mathbb{E}_{A_+}[\mathrm{Ext}] \geq \alpha + (\varepsilon/(1+\varepsilon)) \cdot \alpha(1-\alpha)$.*

To derive the theorem from the claim, assume that $\mathbb{E}[\mathrm{Ext}] \geq \alpha = 1/2 - \varepsilon/10$ with respect to every strategy. By Claim 11 there must then exist a strategy for which

$$\mathbb{E}[\mathrm{Ext}] \geq \frac{1}{2} - \frac{\varepsilon}{10} + \frac{\varepsilon}{1+\varepsilon} \cdot \frac{1 - \varepsilon^2/100}{4}$$

which is at least $1/2 + \varepsilon/10$. ◀

Beigi, Etesami, and Gohari prove Claim 11 under the stronger assumption that NK fails. Their analysis proves the lower bound $\mathbb{E}_{A_+}[\mathrm{Ext}] \geq \alpha + \varepsilon f(\alpha)$ for a more general class of functions f [3, Lemma 10]. The form that we choose is convenient for balancing the simultaneous requirements on mean and variance.

Proof of Claim 11. We prove the claim by induction on n. When $n = 0$ the claim holds by checking the cases $\mathrm{Ext} = 0$ and $\mathrm{Ext} = 1$. We now assume it holds for $n-1$ and prove it for n. Fix A_- to be the strategy that minimizes $\mathbb{E}_{A_-}[\mathrm{Ext}]$, that is at least α. Let d_- be the choice of the first die in this strategy. Then

$$\alpha \leq \mathbb{E}_{d_-}[\alpha_F],$$

where α_f is the conditional expectation of Ext given the first outcome being f, under strategy A_-.

We now describe the strategy A_+. By $\overline{\mathrm{MVR}(\varepsilon)}$ applied to the function $\psi(f) = \alpha_f - \alpha$, there exists a die d_+ such that

$$\mathbb{E}_{d_+}[\alpha_F - \alpha] \geq \varepsilon \mathrm{Var}_{d_+}[\alpha_F]. \tag{2}$$

The adversary A_+ tosses this die first. She then plays the strategy that maximizes $\mathbb{E}_{A_+}[\mathrm{Ext}]$ conditioned on the outcome of the first die. By our inductive assumption, the conditional expectation of Ext when A_+ is played, given the first outcome is f, must be at least $\alpha_f + (\varepsilon/(1+\varepsilon)) \cdot \alpha_f(1-\alpha_f)$ so that

$$\mathbb{E}_{A_+}[\mathrm{Ext}] \geq \mathbb{E}_{d_+}\Big[\alpha_F + \frac{\varepsilon}{1+\varepsilon} \cdot \alpha_F(1-\alpha_F)\Big].$$

We can write

$$\begin{aligned}\mathbb{E}_{d_+}\Big[\alpha_F + \frac{\varepsilon}{1+\varepsilon} \cdot \alpha_F(1-\alpha_F)\Big] - \Big(\alpha + \frac{\varepsilon}{1+\varepsilon} \cdot \alpha(1-\alpha)\Big) \\ = \Big(1 + \frac{\varepsilon}{1+\varepsilon}\Big)\mathbb{E}_{d_+}[\alpha_F - \alpha] - \frac{\varepsilon}{1+\varepsilon}\mathrm{Var}_{d_+}[\alpha_F] - \frac{\varepsilon}{1+\varepsilon}\big(\mathbb{E}_{d_+}[\alpha_F]^2 - \alpha^2\big). \quad (3)\end{aligned}$$

For the last term we have the upper bound

$$\mathbb{E}_{d_+}[\alpha_F]^2 - \alpha^2 = \mathbb{E}_{d_+}[\alpha_F + \alpha] \cdot \mathbb{E}_{d_+}[\alpha_F - \alpha] \leq 2\mathbb{E}_{d_+}[\alpha_F - \alpha],$$

since all the αs are between zero and one, and the second term is non-negative because $\mathbb{E}_{d_+}[\alpha_F] \geq \mathbb{E}_{d_-}[\alpha_F] \geq \alpha$ by the minimality of d_-. We can therefore lower bound the left hand size of (3) by

$$\Big(1 - \frac{\varepsilon}{1+\varepsilon}\Big)\mathbb{E}_{d_+}[\alpha_F - \alpha] - \frac{\varepsilon}{1+\varepsilon}\mathrm{Var}_{d_+}[\alpha_F].$$

By (2) this must be non-negative. It follows that $\mathbb{E}_{A_+}[\mathrm{Ext}]$ is at least $\alpha + (\varepsilon/(1+\varepsilon))\alpha(1-\alpha)$, concluding the inductive step. ◀

2.3 Proof of Theorem 4

▶ **Proposition 12.** *The following conditions are equivalent for a GSV source $(\mathcal{F}, \mathcal{D})$:*

1. *For all $\varepsilon > 0$, $(\mathcal{F}, \mathcal{D})$ satisfies* MVR(ε)*: There exists a $\psi : \mathcal{F} \to [-1, 1]$ such that for all dice d, $|\mathbb{E}_d[\psi(F)]| < \varepsilon \mathrm{Var}_d[\psi(F)]$.*
2. *There exists a constant C such that for sufficiently small $\varepsilon > 0$, there exists a $\psi : \mathcal{F} \to [-1, 1]$ such that for all dice d, $|\mathbb{E}_d[\psi(F)]| < \varepsilon \mathrm{Var}_d[\psi(F)]$ and $\mathrm{Var}_d[\psi(F)] \geq \varepsilon^C$.*
3. *$(\mathcal{F}, \mathcal{D})$ satisfies* HNK*.*

Proof. We will show that 1 implies 3 and 3 implies 2. This will establish equivalence as 2 is a stronger condition than 1.

1 implies 3: Assume that $(\mathcal{F}, \mathcal{D})$ satisfies MVR(ε). This condition is hereditary, namely if it holds for $(\mathcal{F}, \mathcal{D})$ then it holds for all $(\mathcal{F}', \mathcal{D}')$ in the assumption of HNK. So in proving 3, we may and will assume, without loss of generality, that $(\mathcal{F}', \mathcal{D}') = (\mathcal{F}, \mathcal{D})$. We will moreover assume (by scaling and flipping sign if necessary) that ψ attains the value 1.

Now consider an infinite decreasing sequence (ε_k) that converges to zero. By assumption, for every k there exists a ψ_k such that $|\mathbb{E}_d[\psi_k(F)]| < \varepsilon_k \mathrm{Var}_d[\psi_k(F)]$. By the pigeonhole principle there must exist a face f for which the set of indices $K = \{k \colon \psi_k(f) = 1\}$ is infinite. By compactness of $[-1, 1]^{\mathcal{F}}$ there must exist an infinite subset $K' \subseteq K$ for which the subsequence ψ_k over $k \in K'$ converges to a limit ψ. Then ψ is nonzero as $\psi(f)$ must equal one. On the other hand, for every $\varepsilon > 0$ there exists a sufficiently large $k \in K'$ such that for every die d,

$$|\mathbb{E}_d[\psi(F)]| \leq |\mathbb{E}_d[\psi_k]| + \varepsilon \leq \varepsilon \mathrm{Var}_d[\psi_k(F)] + \varepsilon,$$

so $\mathbb{E}_d[\psi(F)]$ must equal zero for every d.

3 implies 2: The proof is by strong induction on the number of dice $|\mathcal{D}|$ with $C = 3 \cdot 2^{|\mathcal{D}|} - 3$. In the base case $|\mathcal{D}| = 1$, all faces must be assigned nonzero probability by the unique die d. Take any witness ψ for HNK. Then $\mathbb{E}_d[\psi(F)] = 0$, but ψ must take nonzero value on at least one of the faces, so $\mathrm{Var}_d[\psi(F)] > 0$. Condition 2 is then satisfied for sufficiently small $\varepsilon > 0$.

For the inductive step, take any ψ that is a witness for HNK with respect to the whole source $(\mathcal{F}, \mathcal{D})$. Let $\mathcal{D}'$ be the subset of dice d such that $\mathrm{Var}_d[\psi(F)] = 0$ and v be the minimum of $\mathrm{Var}_d[\psi(F)]$ over $d \notin \mathcal{D}'$. Then $\mathcal{D}'$ is a proper subset of $\mathcal{D}$ (otherwise, there is a face that is assigned no probability by any die). If $\mathcal{D}'$ is empty, condition 2 follows by the same argument as in the base case. If not, then by the inductive hypothesis we can choose $\psi' \colon \mathcal{F}' \to [-1, 1]$ such that

$$|\mathbb{E}_d[\psi'(F)]| < (v\varepsilon^2/8) \cdot \mathrm{Var}_d[\psi(F)] \quad \text{and} \quad \mathrm{Var}_d[\psi(F)] \geq (v\varepsilon^2/8)^{3 \cdot 2^{|\mathcal{D}'|} - 3}. \tag{4}$$

We will show that the function $\phi = \psi + (v\varepsilon/8) \cdot \psi'$ satisfies the conclusion of condition 2. Here, ψ' is naturally extended as a function on $\mathcal{F}$ by assigning zero on all inputs in $\mathcal{F} \setminus \mathcal{F}'$. The proof is by cases.

If $d \in \mathcal{D}'$, then $\mathbb{E}_d[\phi(F)] = (v\varepsilon/8)\mathbb{E}_d[\psi'(F)]$, while $\mathrm{Var}_d[\phi(F)] = (v\varepsilon/8)^2 \mathrm{Var}_d[\psi'(F)]$. From these two equalities and (4) it follows that $\mathbb{E}_d[\phi(F)] < \varepsilon \mathrm{Var}_d[\phi(F)]$. On the other hand, $\mathrm{Var}_d[\phi(F)] \geq (v\varepsilon/8)^2 \cdot (v\varepsilon^2/8)^{3 \cdot 2^{|\mathcal{D}'|} - 3} \geq \varepsilon^{3 \cdot 2^{|\mathcal{D}|} - 3}$ for sufficiently small ε.

If $d \notin \mathcal{D}'$, then $|\mathbb{E}_d[\phi(F)]| \leq (v\varepsilon/8)|\mathbb{E}_d[\psi'(F)]| \leq v\varepsilon/8$, while

$$\begin{aligned}
\mathrm{Var}_d[\psi(F)] &\geq \mathrm{Var}_d[\psi'(F)] - 2|\mathrm{Cov}_d[\psi(F), (v\varepsilon/8) \cdot \psi'(F)]| \\
&= \mathrm{Var}_d[\psi'(F)] - \frac{v\varepsilon}{4} \cdot |\mathrm{Cov}_d[\psi(F), \psi'(F)]| \\
&\geq \mathrm{Var}_d[\psi'(F)] - \frac{v\varepsilon}{2} \\
&\geq \frac{v}{2},
\end{aligned}$$

where the last inequality follows from our definition of v. In particular, $\mathrm{Var}_d[\psi(F)] \geq \varepsilon^{3 \cdot 2^{|\mathcal{D}|} - 3}$ for sufficiently small ε. On the other hand, $|\mathbb{E}_d[\psi(F)]| \leq v\varepsilon/8 \leq (\varepsilon/4) \cdot \mathrm{Var}_d[\psi(F)]$, as desired. ◀

Proof of Theorem 4. If $(\mathcal{F}, \mathcal{D})$ satisfies HNK, then it also satisfies condition 2 of Proposition 12. By Proposition 7, $(\mathcal{F}, \mathcal{D})$ is extractable with error $O(\sqrt{\varepsilon}) + n/\varepsilon^{C+1}$. The forward direction follows by setting $n = \varepsilon^{C+1.5}$.

For the reverse direction, if $(\mathcal{F}, \mathcal{D})$ fails to satisfy HNK, by Proposition 12, then it also fails to satisfy MVR(ε) for some $\varepsilon > 0$. So by Proposition 10 it is not extractable. ◀

Alternatively, the reverse direction of Theorem 4 can be derived from Theorem 6 of [3] because if $(\mathcal{F}, \mathcal{D})$ fails (NHK) then it contains some $(\mathcal{F}', \mathcal{D}')$ which fails (NK).

3 Randomness-efficient extraction

In this Section we outline the proof of Theorem 5. The implication $1 \to 3$ in Theorem 5 is given by the following Proposition:

▶ **Proposition 13.** *For every $\varepsilon > 0$ and m, every GSV source that satisfies (NK$^+$) is extractable with error ε and output length m from $O((\log(1/\varepsilon) + m)/v^2)$ samples in time $\min\{O(m2^m \cdot n), n^{O(|\mathcal{F}|)}\}$, where v is the minimum of $\mathrm{Var}_d[\psi(F)]$ over all $d \in \mathcal{D}$.*

The case $m = 1$ is proved in Appendix A. The full proof is given in [1].

Implication $3 \to 2$ in Theorem 5 holds trivially. The remaining implication $2 \to 1$ follows readily from Propositions 15 and 16 below. These refer to an analytic condition that characterizes randomness-efficient extractability called the mean-variance divergence (MVD) condition, which can be viewed as the suitable analogue of the MVR condition from Section 2.

▶ **Definition 14.** A GSV source $(\mathcal{F}, \mathcal{D})$ satisfies the *Mean-Variance Divergence condition* $\mathrm{MVD}(\varepsilon, \delta)$ if there exists a function $\psi : \mathcal{F} \to [-1, 1]$ such that for every die $d \in \mathcal{D}$,

$$\left|\mathbb{E}_d[\psi(F)]\right| < \epsilon(\mathrm{Var}_d[\psi(F)] - \delta). \tag{MVD}$$

▶ **Proposition 15.** *If $(\mathcal{F}, \mathcal{D})$ fails (NK^+) then there exists a constant C such that for every $\varepsilon > 0$, $(\mathcal{F}, \mathcal{D})$ fails $\mathrm{MVD}(\varepsilon, C\varepsilon^2)$.*

▶ **Proposition 16.** *If $(\mathcal{F}, \mathcal{D})$ fails $\mathrm{MVD}(\varepsilon, \delta)$ then every extractor with error $\varepsilon/20$ for $(\mathcal{F}, \mathcal{D})$ requires $1/8\delta$ samples, assuming $\varepsilon > 0$ is sufficiently small.*

The proofs are given in Appendix B.

4 Open Questions

In this work, we completely classify GSV sources in terms of their extractability and give optimal deterministic extractors for GSV sources. We point out the following questions for further investigation:

- Let c be the smallest constant for which there exists a non-NK^+ source that is extractable from $O(1/\varepsilon^c)$ samples. Example E2 gives the upper bound $c \leq 7$.[5] Theorem 5 shows that $c \geq 2$. What is the value of c?
- The number of required samples in Theorem 4 is of the form $\varepsilon^{-O(2^{|\mathcal{D}|})}$, where $|\mathcal{D}|$ is the number of dice (see the proof of Proposition 12). Is this exponential dependence in $|\mathcal{D}|$ necessary?
- The multi-bit extractor in Theorem 5 runs in time $\min(nm2^m, n^{O(|\mathcal{F}|)})$. Can the dependence on the number of faces be improved, possibly by applying known seeded extraction algorithms?
- Proposition 7 states that sources satisfying condition $\mathrm{MVR}(\varepsilon)$ admit extraction with error $O(\sqrt{\varepsilon})$, while by Proposition 10 extraction error $\Omega(\varepsilon)$ is necessary. Can this quadratic gap be narrowed?
- It would be interesting to investigate extensions to infinite faces and/or dice.

References

1 Salman Beigi, Andrej Bogdanov, Omid Etesami, and Siyao Guo. Complete classification of generalized Santha-Vazirani sources. Technical Report TR17-136, Electronic Colloquium on Computational Complexity, 2017.

2 Salman Beigi, Omid Etesami, and Amin Gohari. Deterministic randomness extraction from generalized and distributed santha-vazirani sources. In *International Colloquium on Automata, Languages, and Programming*, pages 143–154. Springer, 2015.

[5] This source satisfies $\mathrm{MVR}(\varepsilon)$ with minimum variance ε^2 for every ε (with witness $\psi = (\varepsilon, -\varepsilon, 1, -1)$), so $O(1/\varepsilon^7)$ samples are sufficient for extraction error ε by Proposition 7.

3 Salman Beigi, Omid Etesami, and Amin Gohari. Deterministic randomness extraction from generalized and distributed santha-vazirani sources. *SIAM Journal on Computing*, 46(1):1–36, 2017.

4 Iddo Bentov, Ariel Gabizon, and David Zuckerman. Bitcoin beacon. *arXiv preprint arXiv:1605.04559*, 2016.

5 Jean Bourgain. More on the sum-product phenomenon in prime fields and its applications. *International Journal of Number Theory*, 1(01):1–32, 2005.

6 Jean Bourgain. On the construction of affine extractors. *GAFA Geometric And Functional Analysis*, 17(1):33–57, 2007.

7 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, Cambridge, MA, USA, June 18-21, 2016*, pages 670–683, 2016. doi:10.1145/2897518.2897528.

8 Benny Chor and Oded Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM Journal on Computing*, 17(2):230–261, 1988.

9 Yevgeniy Dodis, Shien Jin Ong, Manoj Prabhakaran, and Amit Sahai. On the (im)possibility of cryptography with imperfect randomness. In *45th Symposium on Foundations of Computer Science (FOCS 2004), 17-19 October 2004, Rome, Italy, Proceedings*, pages 196–205, 2004. doi:10.1109/FOCS.2004.44.

10 Zeev Dvir. Extractors for varieties. *Computational complexity*, 21(4):515–572, 2012.

11 Zeev Dvir, Ariel Gabizon, and Avi Wigderson. Extractors and rank extractors for polynomial sources. *Computational Complexity*, 18(1):1–58, 2009.

12 Ariel Gabizon. Deterministic extractors for affine sources over large fields. In *Deterministic Extraction from Weak Random Sources*, pages 33–53. Springer, 2011.

13 Saeed Mahloujifar, Dimitrios I Diochnos, and Mohammad Mahmoody. Learning under p-tampering attacks. *arXiv preprint arXiv:1711.03707*, 2017.

14 Saeed Mahloujifar and Mohammad Mahmoody. Blockwise p-tampering attacks on cryptographic primitives, extractors, and learners. In *Theory of Cryptography Conference*, pages 245–279. Springer, 2017.

15 Jaikumar Radhakrishnan and Amnon Ta-Shma. Bounds for dispersers, extractors, and depth-two superconcentrators. *SIAM J. Discrete Math.*, 13(1):2–24, 2000. doi:10.1137/S0895480197329508.

16 Miklos Santha and Umesh V. Vazirani. Generating quasi-random sequences from semi-random sources. *J. Comput. Syst. Sci.*, 33(1):75–87, 1986. doi:10.1016/0022-0000(86)90044-9.

17 Salil P Vadhan. *Pseudorandomness*, volume 56. Now, 2012.

A An optimal bit extractor: Proof of Proposition 13 for $m = 1$

Define random variables $Z_0, \ldots, Z_n$ by $Z_0 = 0$ and $Z_t = Z_{t-1} + (\psi(F_t)/2) \cdot (1 - |Z_{t-1}|)$ where F_t $(1 \le t \le n)$ is the t-th output of the GSV source. The extractor outputs the sign of Z_n.

Under (NK$^+$) the sequence (Z_t) is still a martingale so that the expectation of Z_n is 0. But now the range of Z_t is restricted to the open interval $(-1, 1)$. To prove that the sign of Z_n has small bias, we begin by showing that the logarithm of $1/D_t$, on average, grows by a constant in every step where $D_t = 1 - |Z_t|$ is the distance between Z_t and its sign. Then we will use to argue the expectation of D_n is exponentially small. This fact together with $\mathbb{E}[Z_n] = 0$ allows us to conclude that the sign of Z_n is exponentially close to being unbiased.

▶ **Claim 17.** $\mathbb{E}[\ln(1/D_t) - \ln(1/D_{t-1}) \mid D_{<t}] \ge v/24$.

Proof. Using the identity $|x| = \text{sign}(x) \cdot x$, we upper bound D_t by

$$\begin{aligned} D_t &= 1 - |Z_t| \le 1 - \text{sign}(Z_{t-1}) \cdot Z_t \\ &= 1 - \text{sign}(Z_{t-1})\Big(Z_{t-1} + \frac{\psi(F_t)}{2} \cdot D_{t-1}\Big) = \Big(1 - \text{sign}(Z_{t-1}) \cdot \frac{\psi(F_t)}{2}\Big) \cdot D_{t-1}. \end{aligned} \tag{5}$$

Therefore,

$$\begin{aligned} \mathbb{E}\Big[\ln \frac{D_{t-1}}{D_t} \;\Big|\; D_{<t}\Big] &\ge \mathbb{E}\Big[-\ln\Big(1 - \text{sign}(Z_{t-1}) \cdot \frac{\psi(F_t)}{2}\Big) \;\Big|\; D_{<t}\Big] \\ &\ge \mathbb{E}\Big[\text{sign}(Z_{t-1}) \cdot \frac{\psi(F_t)}{2} + \frac{1}{6}\Big(\text{sign}(Z_{t-1}) \cdot \frac{\psi(F_t)}{2}\Big)^2 \;\Big|\; D_{<t}\Big] \\ &= \mathbb{E}\Big[\text{sign}(Z_{t-1}) \cdot \frac{\psi(F_t)}{2} \;\Big|\; D_1, \ldots, D_{t-1}\Big] + \frac{1}{6}\mathbb{E}\Big[\Big(\frac{\psi(F_t)}{2}\Big)^2 \;\Big|\; D_{<t}\Big] \\ &\ge 0 + v/24. \end{aligned}$$

The inequalities follow from the positivity of the (D_i)s, the identity $-\ln(1-x) \ge x + x^2/6$, the boundedness of ψ, and the NK$^+$ assumption, respectively. ◀

Let $X_t = \ln(1/D_t) - (vt/24)$. By Claim 17, $\mathbb{E}[X_t | D_{<t}] \ge X_{t-1}$ so that the sequence (X_t) is a sub-martingale with respect to (D_t). By (5) and the triangle inequality

$$|X_t - X_{t-1}| \le |\ln D_t/D_{t-1}| + v/24 \le \max\{|\ln 1/2|, |\ln 3/2|\} + v/24 = \ln 2 + v/24$$

By Azuma's inequality,

$$\Pr[D_n \ge e^{-vn/48}] = \Pr[X_n \le -vn/48] \le e^{-(vn/48)^2/2n(\ln 2 + v/24)^2} = 2^{-\Omega(v^2 n)}.$$

Finally by the triangle inequality and the law of conditional expectations,

$$|\mathbb{E}[\text{sign}(Z_n)]| \le \mathbb{E}[D_n] + |\mathbb{E}[Z_n]| \le 2^{-vn/48} + \Pr[D_n \ge e^{-vn/48}] + |\mathbb{E}[Z_n]| = 2^{-\Omega(v^2 n)}.$$

B A lower bound on the quality of extraction

The *kernel* of GSV source $(\mathcal{F}, \mathcal{D})$, denoted by $\text{Ker}\,\mathcal{D}$, is the set of all $\psi\colon \mathcal{F} \to \mathbb{R}$ such that $\mathbb{E}_d[\psi_d(F)] = 0$ for all dice $d \in \mathcal{D}$.

▶ **Proposition 18.** *A GSV source $(\mathcal{F}, \mathcal{D})$ satisfies (NK$^+$) if and only if for every die $d \in \mathcal{D}$ there exists a function $\psi_d \in \text{Ker}\,\mathcal{D}$ that is not constant on the support of d.*

Proof of Proposition 18. The forward direction follows by setting all ψ_d to equal the witness ψ for the (NK$^+$) condition. For the reverse direction, let $\psi = \sum_{d \in \mathcal{D}} N_d \psi_d$ where N_d are independent random variables, each uniformly distributed over some finite set $\mathcal{N} \subseteq \mathbb{R}$ of size more than $|D|$. By linearity, ψ is in $\text{Ker}\,\mathcal{D}$. Moreover, for each die d and each possible choice of the values $N_{d'}$ for $d' \ne d$, the sum $\sum N_d \psi_d$ can be constant on the support of d for at most one choice of N_d (for if two such choices existed then ψ_d itself must be constant on the support of d). Therefore, ψ is constant on d with probability at most $1/|\mathcal{N}|$. Since $|\mathcal{N}| > |\mathcal{D}|$, the existence of an (NK$^+$) witness ψ follows from the union bound. ◀

▶ **Claim 19.** *If (NK$^+$) fails for GSV source $(\mathcal{F}, \mathcal{D})$ then there exists a die $d \in \mathcal{D}$ such that for every pair of faces f^*, f_* in the support of d there exists a function $\beta\colon \mathcal{D} \to \mathbb{R}$ such that for all functions $\psi\colon \mathcal{F} \to \mathbb{R}$,*

$$\psi(f^*) - \psi(f_*) = \sum_{d' \in \mathcal{D}} \beta(d') \cdot \mathbb{E}_{d'}[\psi(F)]. \tag{6}$$

Proof. If $f^* = f_*$ the conclusion holds with $\beta = 0$. Otherwise, let $\mathcal{C}_d$ denote the linear space of functions that are constant on the support of die d. By Proposition 18, if (NK^+) fails then there exists a die d for which all functions $\psi \in \operatorname{Ker} \mathcal{D}$ also belong to $\mathcal{C}_d$, i.e., $\operatorname{Ker} \mathcal{D} \subseteq \mathcal{C}_d$. Then $\mathcal{C}_d^\perp \subseteq (\operatorname{Ker} \mathcal{D})^\perp$, where $\perp$ indicates the dual subspace. The space $(\operatorname{Ker} \mathcal{D})^\perp$ is the span of the probability mass functions pmf_d of all the dice. Therefore every $\phi \in \mathcal{C}_d^\perp$ can be written as a linear combination

$$\phi = \sum_{d \in \mathcal{D}} \beta(d) \cdot \mathrm{pmf}_d.$$

Then for every $\psi \colon \mathcal{F} \to \mathbb{R}$,

$$\sum_{f \in \mathcal{F}} \phi(f) \cdot \psi(f) = \sum_{d \in \mathcal{D}, f \in \mathcal{F}} \beta(d) \cdot \mathrm{pmf}_d(f) \cdot \psi(f) = \sum_{d \in \mathcal{D}} \beta(d) \cdot \mathbb{E}_d[\psi(F)].$$

The claim follows by specializing ϕ to the function that takes value 1 on f^*, -1 on f_*, and 0 elsewhere. This function is dual to $\mathcal{C}_d$. ◀

Proof of Proposition 15. Assume $(\mathcal{F}, \mathcal{D})$ fails (NK^+). Let d be the die stipulated by Claim 19 and C be the maximum of $(\sum_{d' \in \mathcal{D}} |\beta(d')|)^2$ over all pairs of faces f^*, f_* in the support of d.

Towards a contradiction suppose that $(\mathcal{F}, \mathcal{D})$ satisfies $\mathrm{MVD}(\varepsilon, \delta)$. Then the witness $\psi \colon \mathcal{F} \to [-1, 1]$ for $\mathrm{MVD}(\varepsilon, \delta)$ must satisfy the conditions $\mathrm{Var}_d[\psi(F)] > \delta$ and $|\mathbb{E}_{d'}[\psi(F)]| < \varepsilon \mathrm{Var}_{d'}[\psi(F)]$ for all dice $d' \in \mathcal{D}$. Let f^* and f_* be faces in the support of d that maximize and minimize the value of ψ, respectively. By Claim 19, relation (6) holds for some β that may depend on f^* and f_* but not on ψ. Then

$$\begin{aligned}\sqrt{\delta} < \sqrt{\mathrm{Var}_d[\psi(F)]} \le \psi(f^*) - \psi(f_*) &= \sum_{d' \in \mathcal{D}} \beta(d') \cdot \mathbb{E}_{d'}[\psi(F)] \\ &\le \sum_{d' \in \mathcal{D}} |\beta(d')| \cdot |\mathbb{E}_{d'}[\psi(F)]| < \sum_{d' \in \mathcal{D}} |\beta(d')| \cdot \varepsilon \mathrm{Var}_{d'}[\psi(F)] \le \sqrt{C}\varepsilon,\end{aligned}$$

where the last inequality follows from the definition of C and the boundedness of ψ. Therefore $\mathrm{MVD}(\varepsilon, \delta)$ fails for $\delta = C\varepsilon^2$. ◀

Proof of Proposition 16. The proof is a direct extension of the proof of Proposition 10. The main technical tool is the following claim:

▶ **Claim 20.** *For every extractor* $\mathrm{Ext} \colon \mathcal{F}^n \to \{0, 1\}$*, and every* $0 \le \alpha \le 1$*, if* $\mathbb{E}_{A_-}[\mathrm{Ext}] \ge \alpha$ *for every strategy* A_-*, then there exists a strategy* A_+ *for which*

$$\mathbb{E}_{A_+}[\mathrm{Ext}] \ge \alpha + \frac{\varepsilon}{1+\varepsilon} \cdot \big(\alpha(1-\alpha) - \delta n\big).$$

The proof of Claim 20 is a notationally intensive direct extension of the proof of Claim 11. We omit the details.

By Claim 20 it follows that for every $\varepsilon > 0$, if no strategy A_- has error less than $\alpha = 1/2 - \varepsilon/20$ against Ext then there exists a strategy A_+ with

$$\mathbb{E}_{A_+}[\mathrm{Ext}] \ge \frac{1}{2} - \frac{\varepsilon}{20} + \frac{\varepsilon}{1+\varepsilon} \cdot \Big(\frac{1 - \varepsilon^2/400}{4} - \frac{1}{8}\Big),$$

which is at least $1/2 + \varepsilon/20$ for sufficiently small ε. ◀

Adaptive Lower Bound for Testing Monotonicity on the Line

Aleksandrs Belovs[1]
Faculty of Computing, University of Latvia, Raina bulvaris 19, Riga, Latvia.
aleksandrs.belovs@lu.lv

Abstract

In the property testing model, the task is to distinguish objects possessing some property from the objects that are far from it. One of such properties is monotonicity, when the objects are functions from one poset to another. This is an active area of research. In this paper we study query complexity of ε-testing monotonicity of a function $f\colon [n] \to [r]$. All our lower bounds are for adaptive two-sided testers.

- We prove a nearly tight lower bound for this problem in terms of r. The bound is $\Omega\big(\frac{\log r}{\log\log r}\big)$ when $\varepsilon = 1/2$. No previous satisfactory lower bound in terms of r was known.
- We completely characterise query complexity of this problem in terms of n for smaller values of ε. The complexity is $\Theta\big(\varepsilon^{-1}\log(\varepsilon n)\big)$. Apart from giving the lower bound, this improves on the best known upper bound.

Finally, we give an alternative proof of the $\Omega(\varepsilon^{-1}d\log n - \varepsilon^{-1}\log\varepsilon^{-1})$ lower bound for testing monotonicity on the hypergrid $[n]^d$ due to Chakrabarty and Seshadhri (RANDOM'13).

2012 ACM Subject Classification Theory of computation → Lower bounds and information complexity

Keywords and phrases property testing, monotonicity on the line, monotonicity on the hypergrid

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.31

Funding This research is supported by the ERDF project number 1.1.1.2/I/16/113.

Acknowledgements I am grateful to Eric Blais for introducing me to this problem, and for his encouragement to work on it, as well as for pointing out Ref. [18]. I would like to thank Dmitry Gavinsky, Ansis Rosmanis, and Ronald de Wolf for helpful discussions, and anonymous referees for their suggestions. The construction of Theorem 7 is due to Ronald de Wolf. Part of this work was done while visiting Institute of Mathematics of the Czech Academy of Sciences in Prague, and Centre for Quantum Technologies in Singapore. I would like to thank Dmitry Gavinsky, Pavel Pudlák, and Miklos Santha for hospitality.

1 Introduction

The framework of property testing was formulated by Rubinfeld and Sudan [19] and Goldreich et al. [16]. A property testing problem is specified by a *property* $\mathcal{P}$, which is a class of functions mapping some finite set D into some finite set R, and *proximity parameter* ε, which is a real number between 0 and 1. An *ε-tester* is a bounded-error randomised query algorithm which, given oracle access to a function $f\colon D \to R$, distinguishes between the case when f belongs to $\mathcal{P}$ and the case when f is ε-far from $\mathcal{P}$. The latter means that any function $g \in \mathcal{P}$ differs from f on at least ε fraction of the points in the domain D. The usual complexity measure

[1] supported by the ERDF project number 1.1.1.2/I/16/113

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018).
Editors: Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer; Article No. 31; pp. 31:1–31:10

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

is the number of queries to the function f. A tester is with *1-sided* error if it always accepts a function f in $\mathcal{P}$. A tester is *non-adaptive* if its queries do not depend on the responses received to the previous queries. The most general tester is adaptive with 2-sided error, which we will implicitly assume in this paper.

When both the domain D and the range R are partially ordered sets, a natural property to consider is that of *monotonicity*. A function $f\colon D \to R$ is called *monotone* if $x \leq y$ implies $f(x) \leq f(y)$ for all $x, y \in D$. Usually it is assumed that R is a totally ordered set. In this case, the property $\mathcal{P}$ consists of all monotone functions from D to R. The problem of testing monotonicity was explicitly formulated and studied by Goldreich et al. [15] in the case when D is the Boolean hypercube $\{0,1\}^d$ and $R = \{0,1\}$. This is an active area of research as exemplified by the papers [11, 14, 7, 6, 9, 8, 17, 1, 2, 10].

However, slightly earlier than Goldreich et al., Ergün et al. studied the same problem for functions $f\colon [n] \to [r]$. Ergün et al. called it *spot-checker for sorting*, but now this problem is generally known as *monotonicity testing on the line*, the line being the totally ordered set $[n]$. This problem is the main focus of this paper. Although this problem is arguably simpler than testing monotonicity on the hypercube, it seems more natural and important from the practical point of view. Ergün et al. mention that their algorithm can be used in software quality assurance by providing a very fast verification procedure that checks whether a presumably sorted array is indeed sorted.

A related problem is that of testing monotonicity on the *hypergrid*. A hypergrid is a set $[n]^d$ of d-tuples with elements in $[n]$. For two d-tuples $x = (x_1, \ldots, x_d)$ and $y = (y_1, \ldots, y_d)$, we have $x \leq y$ iff $x_i \leq y_i$ for all i. We are interested in functions from $[n]^d$ to $[r]$ for some positive integers n, d and r. Clearly, this is a generalisation of both monotonicity testing on the line (when $d = 1$) and on the hypercube (when $n = 2$). These problems are closely related, so it is important to consider all of them when discussing prior work.

1.1 Prior work

We proceed with a brief discussion of previous results related to our paper. Let us start with the upper bounds. As mentioned above, the problem of testing monotonicity on the line was first considered by Ergün et al. [12], who gave an ε-tester with complexity $O\big(\varepsilon^{-1} \log n\big)$. Concerning the case of functions from the hypercube $\{0,1\}^d$ to arbitrary $[r]$, Goldreich et al. [15] proposed the *edge tester*, and Chakrabarty and Seshadhri [7] proved that it has query complexity $O(d/\varepsilon)$. In the latter paper, an ε-tester for testing monotonicity on the hypergrid $[n]^d$ with complexity $O(\varepsilon^{-1} d \log n)$ was also constructed.

Now let us turn to the lower bounds. Ergün et al. [12] proved a lower bound[2] of $\Omega(\log n)$ for testing monotonicity on the line in the so-called *comparison-based* model. In this model, each query of the tester may depend only on the order relations between the responses to the previous queries, but not on the values of the responses themselves.[3] Fischer [13] proved that any lower bound for a monotonicity testing problem in the comparison-based model implies the same lower bound in the usual value-based model. This immediately gives an $\Omega(\log n)$ lower bound for testing monotonicity on the line, matching the upper bound by Ergün et al. [12]. Chakrabarty and Seshadhri [5] used the same technique to prove a lower bound of

[2] If a lower bound does not state dependence on ε, it is assumed that the lower bound holds for some choice of $\varepsilon = \Omega(1)$.

[3] Note that this does *not* mean that the tester asks queries of the form $f(x) \stackrel{?}{\leq} f(y)$. The query is still an input x, and the tester learns about the order relations between $f(x)$ and $f(y)$ for *all* previously queried y's.

$\Omega(\varepsilon^{-1}d\log n - \varepsilon^{-1}\log\varepsilon^{-1})$ for testing monotonicity on hypergrids. Unfortunately, Fischer's construction is based on Ramsey theory, which means that, in order for this construction to work, the size of the range, r, has to be really huge.

Large values of r in the above lower bounds may lead one to study complexity of testing monotonicity with respect to the size of the range, r, rather than the size of the domain, n. Moreover, there are two parameters related to the output of the function f: the size of the range (codomain) r, and the size of the image $t = \big|f([n])\big|$: the number of different values attained by the function. Clearly, $t \le r$. This means it is more interesting to prove upper bounds in terms of t and lower bounds in terms of r.

Let us start with the upper bounds. First, it is easy to see that if $r = 2$, then $O(1/\varepsilon)$ queries suffice to ε-test monotonicity on the line. Generalising this observation, Pallavoor et al. [18] constructed an ε-tester for monotonicity on the line with complexity $O(\frac{1}{\varepsilon}\log t)$ for general t, as well as an ε-tester for monotonicity on hypergrids with complexity $O\big(\frac{d}{\varepsilon}\log\frac{d}{\varepsilon}\log t\big)$. Since $t \le n$, the lower bound of $\Omega(\log n)$ due to Fischer [13] implies the lower bound of $\Omega(\log t)$ for all $n \ge t$ and r large enough.[4]

The main prior technique capable of proving strong lower bounds for functions with small range is that of communication complexity. Blais et al. introduced this technique in [3], where it was proven that $\Omega\big(\min\{d, r^2\}\big)$ queries are required to test a function $f\colon \{0,1\}^d \to [r]$ for monotonicity. In a subsequent paper [4] a *non-adaptive* lower bound of $\Omega(d\log n)$ was proven for functions $f\colon [n]^d \to [nd]$ on the hypergrid. In the special case of $d = 1$, this gives a lower bound of $\Omega\big(\log\min\{n, r\}\big)$ for testing monotonicity of a function $f\colon [n] \to [r]$ on the line.

1.2 Our results

Our main contribution is an adaptive lower bound for testing monotonicity on the line. In order not to obstruct our main argument, we first prove the bound for $\varepsilon = \Omega(1)$, and then show how to adapt the construction for smaller values of ε.

▶ **Theorem 1.** *Every adaptive bounded-error $1/2$-tester for monotonicity of a function $f\colon [2^k] \to [k^{3k}]$ has query complexity $\Omega(k)$.*

Unlike [13, 5], we bypass Ramsey theory and construct two explicit distributions of functions that are hard to distinguish by an adaptive algorithm. In terms of the size of the domain, n, this gives the same lower bound of $\Omega(\log n)$ as in Fischer's paper [13], but with vastly reduced range size. A more direct construction can be beneficial for generalisation to other models, like quantum testers, since it is not known how to adapt Fischer's technique to the quantum settings.

In terms of the size of the range, r, this gives a lower bound of of $\Omega\big(\frac{\log r}{\log\log r}\big)$, thus nearly matching the upper bound by Pallavoor et al. [18]. Finally, we get a slightly worse estimate (in terms of the range size) than that of Blais et al. [4] but for *adaptive* testers. We prove Theorem 1 in Section 3.

In Section 4, we consider the case of general ε. First, we show that the construction of Theorem 1 can be used to prove an $\Omega\big(\varepsilon^{-1}\log(\varepsilon n)\big)$ lower bound for ε-testing monotonicity on the line. For large values of ε, this matches the upper bound of $O\big(\varepsilon^{-1}\log n\big)$ due to Ergün et al. [12], but is slightly worse when ε is close to $1/n$. However, we manage to improve the algorithm and prove an upper bound of $O\big(\varepsilon^{-1}\log(\varepsilon n)\big)$ for all $\varepsilon n \ge 2$, thus matching our lower bound. If $\varepsilon n < 2$, the complexity is obviously $\Theta(n)$, hence, this completely resolves the problem of ε-testing monotonicity on the line for all values of n and $\varepsilon \le 1/2$.

[4] The size of the domain of a function $f\colon [n] \to [r]$ can be inflated without changing its distance to monotonicity by replacing f with the function $f'\colon [nk] \to [r]$ given by $f'(x) = f(\lfloor x/k \rfloor)$.

Finally, in Section 5, we show how our construction can be adapted to prove the lower bound $\Omega(\varepsilon^{-1}d\log n - \varepsilon^{-1}\log\varepsilon^{-1})$ for ε-testing monotonicity on the hypergrid $[n]^d$. This coincides with the bound by Chakrabarty and Seshadhri [5], but, again, our construction bypasses Ramsey theory, which results in a vastly reduced range size.

2 Preliminaries

Although this is not standard, it will be convenient for us to denote $[n] = \{0, 1, \ldots, n-1\}$, and $[a..b] = \{a, a+1, \ldots, b-1\}$. Thus, $[n] = [0..n]$, and note that $[a..b]$ does not contain b.

An *assignment* $\alpha\colon S \to [r]$ is a function defined on a subset $S \subseteq [n]$. The *weight* of α is the size of S. We say that f *agrees* with α if $f(x) = \alpha(x)$ for all $x \in S$. This is notated by $f \rightharpoonup \alpha$. The choice of notation is to distinguish it from $f \sim \mu$ which means that f is distributed according to the probability distribution μ.

All logarithms are to the base of 2.

3 Proof of Theorem 1

We will define a probability distribution μ on monotone functions $f\colon [2^k] \to [k^{3k}]$ and a probability distribution ν on functions $g\colon [2^k] \to [k^{3k}]$ that are 1/2-far from monotone. It will be impossible to distinguish these two distributions using fewer than $\Omega(k)$ queries. Let $m = k^3$, so that $r = m^k$.

We define the distribution μ in two different but equivalent ways. First, μ is defined as the last member in an inductively-defined family $\mu_0, \mu_1, \ldots, \mu_k$ of distributions, where μ_i is supported on functions $[2^i] \to [m^i]$. The distribution μ_0 is supported on the only function that maps 0 to 0. Assume that μ_i is already defined and let us define μ_{i+1}. In order to do that, we independently sample f_0 and f_1 from μ_i and a from $[m-1]$. The corresponding function f in μ_{i+1} is given by

$$f(x) = \begin{cases} a \cdot m^i + f_0(x), & \text{if } 0 \le x < 2^i; \\ (a+1)m^i + f_1(x - 2^i), & \text{if } 2^i \le x < 2^{i+1}. \end{cases} \tag{1}$$

For an alternative way of defining μ, let us assume that the argument x is written in binary and the value $f(x)$ in m-ary. We prepend leading zeroes if necessary so that each number has exactly k digits. We enumerate the digits from left to right with the elements of $[k]$, so that the 0-th digit is the most significant one, and the $(k-1)$-st digit is the least significant one. For each binary string s of length strictly less than k, sample an element a_s from $[m-1]$ independently and uniformly at random. The i-th digit of $f(x)$ is defined as $a_s + b$, where s is the prefix of x of length i and b is the i-th bit of x. It is easy to see that both definitions of μ are equivalent, and that any function f from the support of μ is monotone.

The distribution ν is defined as the uniform mixture of the following distributions ν^j for $j \in [k]$. The function $g \sim \nu^j$ is defined as $g(x) = f(x \oplus 2^{k-1-j})$ when f is sampled from μ. Here $\oplus$ denotes the bit-wise XOR function. In other words, the jth bit of the argument is flipped before applying f. Alternatively, we may say that $g \sim \nu^j$ is defined as in the case of μ with the exception that the j-th digit of $g(x)$ is $a_s + (1-b)$ instead of $a_s + b$.

▶ **Claim 2.** *Any function g in the support of ν^j is 1/2-far from monotone.*

Proof. Consider two input strings $x < y$ that differ only in the j-th bit. By the definition of ν^j we have $g(x) > g(y)$, thus, $\{x, y\}$ is a monotonicity-violating pair. We have 2^{k-1} such disjoint pairs, and every monotone function differs from g on at least one element of each pair. ◀

Now we are ready to start with the proof of Theorem 1. Assume towards contradiction that there exists a randomised 1/2-tester $\mathcal{A}$ with query complexity $o(k)$. Using standard error reduction, we may assume that $\mathcal{A}$ errs with probability at most 1/8 on each input. Let λ be the uniform mixture of μ and ν. Clearly, $\mathcal{A}$ errs with probability at most 1/8 on λ. The randomised algorithm $\mathcal{A}$ can be defined as a probability distribution on deterministic query algorithms of the same query complexity, hence, one of the deterministic algorithms in the support of $\mathcal{A}$ errs with probability at most 1/8 on λ. Thus, we may assume $\mathcal{A}$ is a *deterministic* query algorithm. The error probability 1/8 on λ implies that on μ and ν we have

$$\Pr_{f\sim\mu}[\mathcal{A} \text{ accepts } f] \geq \frac{3}{4} \qquad \text{and} \qquad \Pr_{g\sim\nu}[\mathcal{A} \text{ accepts } g] \leq \frac{1}{4}. \tag{2}$$

Also, we may assume that k is large enough, so that the query complexity of the deterministic query algorithm $\mathcal{A}$ is less than $k/2$.

So, $\mathcal{A}$ is a decision tree. Its leaves are specified by assignments in the sense that $\mathcal{A}$ terminates its work on input f in a leaf given by assignment α if and only if f agrees with α. Moreover, the weight of each such α does not exceed the query complexity of $\mathcal{A}$. We partition all these assignments α into two parts as follows. We say that an integer $\ell \in [r]$ is *good* if it contains no digit 0 and no digit $m-1$ (in the m-ary representation as before). Otherwise, ℓ is *bad*. We say that an assignment α is *good* if all the elements in its image are good. Otherwise, α is *bad*. Finally, a leaf of $\mathcal{A}$ is good iff the corresponding assignment is good.

▶ **Claim 3.** *The probability that $\mathcal{A}$ ends its work in a bad leaf when run on an input f sampled from μ is $o(1)$.*

Proof. For each x in the domain of f, the value $f(x)$ is bad only if one of the k elements $a_{s_0}, a_{s_1}, \ldots, a_{s_{k-1}}$ has value 0 or $m-2$, where s_i is the prefix of x of length i. Thus, the probability of $\mathcal{A}$ to terminate in a bad leaf is at most the probability of finding an element a_s with value in $\{0, m-2\}$ with $k/2$ queries, where on each query it is allowed to test the values of k different a_s's. Since the a_s's are independent, and the probability of $a_s \in \{0, m-2\}$ is $2/(m-1)$, we get, using the standard bound on search, that the probability of succeeding is at most $\frac{k}{2} \cdot k \cdot \frac{2}{m-1} = o(1)$. ◀

▶ **Claim 4.** *For each good assignment α of weight at most $k/2$,*

$$\Pr_{f\sim\mu}[f \rightarrowtail \alpha] \leq 2 \cdot \Pr_{g\sim\nu}[g \rightarrowtail \alpha].$$

Before we start with the proof of this claim, let us show how Theorem 1 follows from Claims 3 and 4. In the following, let C be the set of assignments which correspond to the accepting leaves of $\mathcal{A}$, let $B \subseteq C$ be the subset of bad assignments, and $G \subseteq C$ be the subset of good assignments. Then,

$$\begin{aligned}\Pr_{f\sim\mu}[\mathcal{A} \text{ accepts } f] &= \sum_{\alpha\in B} \Pr_{f\sim\mu}[f \rightarrowtail \alpha] + \sum_{\alpha\in G} \Pr_{f\sim\mu}[f \rightarrowtail \alpha] \\ &\leq o(1) + 2\sum_{\alpha\in G} \Pr_{g\sim\nu}[g \rightarrowtail \alpha] \leq o(1) + 2\Pr_{g\sim\nu}[\mathcal{A} \text{ accepts } g],\end{aligned}$$

which is in contradiction with (2) if k is large enough.

It remains to prove good. Consider a good assignment α of weight at most $k/2$. Let S be the domain of α. We say that a pair $x < y$ from S *cuts* an index $j \in [k]$ iff their first j bits agree, and they disagree in the j-th bit. In other words, there exists $a \in \mathbb{Z}$ such that

$$2a \cdot 2^{k-j-1} \leq x < (2a+1) \cdot 2^{k-j-1} \leq y < (2a+2) \cdot 2^{k-j-1}.$$

The assignment α cuts all the indices cut by the pairs in S. good follows from the following two lemmata.

▶ **Lemma 5.** *If a good assignment α does not cut an index j, then*

$$\Pr_{f \sim \mu}[f \rightarrow \alpha] = \Pr_{g \sim \nu^j}[g \rightarrow \alpha].$$

Proof. By induction on i in the definition (1) of μ_i. Let for brevity $j' = k - j - 1$. If $j' < i$, we define ν_i^j as the distribution over the functions $g(x) = f(x \oplus 2^{j'})$ when f is sampled from μ_i. If $j' \geq i$, we define $\nu_i^j = \mu_i$. We prove that

$$\Pr_{f \sim \mu_i}[f \rightarrow \alpha] = \Pr_{g \sim \nu_i^j}[g \rightarrow \alpha] \tag{3}$$

for every good assignment α from $[2^i]$ to $[m^i]$ that does not cut the index j.

The base case $i = 0$ is trivial. (Actually, the statement is trivial for all $i \leq j'$.) Assume (3) is proven for i, and let us prove it for $i+1$. Let S be the domain of α. There are two cases.

First, assume both $S \cap [2^i]$ and $S \cap [2^i..2^{i+1}]$ are non-empty. This means that α cuts $k - i - 1$, hence, $i \neq j'$. Also, we may assume there exists $1 \leq a \leq m-3$ such that $\alpha([2^i]) \subseteq [am^i..(a+1)m^i]$ and $\alpha([2^i..2^{i+1}]) \subseteq [(a+1)m^i..(a+2)m^i]$, since otherwise both sides of (3) are 0. Under these assumptions,

$$\Pr_{f \sim \mu_{i+1}}[f \rightarrow \alpha] = \frac{1}{m-1} \Pr_{f_0 \sim \mu_i}[f_0 \rightarrow \alpha_0] \Pr_{f_1 \sim \mu_i}[f_1 \rightarrow \alpha_1],$$

where f_0 and f_1 are obtained reversely from (1), and α_0 and α_1 are defined similarly: $\alpha_0(i) = \alpha(i) \bmod m^i$ and $\alpha_1(i) = \alpha(i + 2^i) \bmod m^i$ for all $i \in [2^i]$. Both assignments are good, and they do not cut the index j. Similarly, since $i \neq j'$:

$$\Pr_{g \sim \nu_{i+1}^j}[g \rightarrow \alpha] = \frac{1}{m-1} \Pr_{g_0 \sim \nu_i^j}[g_0 \rightarrow \alpha_0] \Pr_{g_1 \sim \nu_i^j}[g_1 \rightarrow \alpha_1].$$

By the inductive assumption, we have the required equality.

Now assume one of $S \cap [2^i]$ and $S \cap [2^i..2^{i+1}]$ is empty. We consider the case $S \subseteq [2^i]$, the second one being similar. Again, we can assume there exists $1 \leq a \leq m-2$ such that $\alpha([2^i]) \subseteq [am^i..(a+1)m^i]$. Then,

$$\Pr_{f \sim \mu_{i+1}}[f \rightarrow \alpha] = \frac{1}{m-1} \Pr_{f_0 \sim \mu_i}[f_0 \rightarrow \alpha_0]$$

and, no matter whether $i = j'$ or not,

$$\Pr_{g \sim \nu_{i+1}^j}[g \rightarrow \alpha] = \frac{1}{m-1} \Pr_{g_0 \sim \nu_i^j}[g_0 \rightarrow \alpha_0],$$

and again we have the required equality by the inductive assumption. ◀

▶ **Lemma 6.** *An assignment α of weight t cuts at most $t-1$ indices in $[k]$.*

Proof. Let S be the domain of α, and let $J \subseteq [k]$ be the set of indices that α cuts.

Let us construct a graph G as follows. Its vertex set is S. For every index $j \in J$ take one arbitrary pair of elements $x, y \in S$ that cuts j and connect x and y by an edge. We say that the edge xy cuts j.

We claim that the graph G is acyclic, from which the statement of the lemma follows. Assume that G contains a simple cycle. Consider an edge xy that cuts the minimal index j on this cycle. Then x and y disagree in the j-th bit. On the other hand, considering the remaining part of the cycle, we see that x and y agree in the j-th bit. A contradiction, hence, G is acyclic. ◀

By cut, there are at least $k/2$ indices not cut by α, and using goodalpha, we have

$$2 \cdot \Pr_{g\sim\nu}[g \rightharpoonup \alpha] \geq \frac{2}{k} \sum_{j:\alpha \text{ does not cut } j} \Pr_{g\sim\nu^j}[g \rightharpoonup \alpha] \geq \frac{2}{k} \cdot \frac{k}{2} \cdot \Pr_{f\sim\mu}[f \rightharpoonup \alpha] = \Pr_{f\sim\mu}[f \rightharpoonup \alpha],$$

proving good.

4 The Case of Small ε

In this section, we briefly describe how the result of Theorem 1 can be extended to arbitrary values of ε, and give an improved version of the algorithm by Ergün et al. [12].

▶ **Theorem 7.** *If $\varepsilon \leq 1/2$, the complexity of ε-testing a function $f\colon [n] \to [r]$ for monotonicity is*

$$\Omega\left(\min\left\{\frac{\log(\varepsilon n)}{\varepsilon}, \frac{\log(\varepsilon r)}{\varepsilon \log\log(\varepsilon r)}\right\}\right).$$

Proof. The proof closely follows that of Theorem 1. We will briefly describe the construction and the proof using the notation of Section 3.

Let ℓ be a positive integer and assume $\varepsilon = 1/(2\ell)$. We will construct probability distributions $\widetilde{\mu}$ and $\widetilde{\nu}$ on functions $f\colon [\ell 2^k] \to [\ell k^{3k}]$ such that all functions in the support of $\widetilde{\mu}$ are monotone, functions in the support of $\widetilde{\nu}$ are ε-far from monotone, and it takes $\Omega(\ell k)$ queries to distinguish $\widetilde{\mu}$ and $\widetilde{\nu}$. Expressing ℓk in terms of ε and n or in terms of ε and r gives the required bound.

Let μ and ν be as in Section 3. For $s \in [\ell]$, independently sample f_s from μ. Define $f \sim \widetilde{\mu}$ as

$$f(s \cdot 2^k + x) = s \cdot k^{3k} + f_s(x) \tag{4}$$

for all $s \in [\ell]$ and $x \in [2^k]$. The distribution $\widetilde{\nu}$ is defined as the uniform mixture of $\widetilde{\nu}^{t,j}$ as t ranges over $[\ell]$ and j over $[k]$. The corresponding function $f \sim \widetilde{\nu}^{t,j}$ is defined as in (4) with exception that f_t is sampled from ν^j instead of μ. It is easy to see that functions in the support of $\widetilde{\mu}$ are monotone, and, using non-monotone, that functions in the support of $\widetilde{\nu}$ are ε-far from monotone.

Informally, it takes $\Omega(\ell k)$ queries to distinguish $\widetilde{\mu}$ and $\widetilde{\nu}$ because we are searching for one non-monotone distribution ν^j among ℓ independent distributions. Formally, we may proceed as follows. Assume towards contradiction that there exists a deterministic query algorithm $\mathcal{A}$ that makes less than $\ell k/4$ queries, accepts $\widetilde{\mu}$ with probability at least $3/4$ and accepts $\widetilde{\nu}$ with probability at most $1/4$.

Using the same reasoning as in bad, the expected number of bad elements found by $\mathcal{A}$ when run on $\widetilde{\mu}$ is $O(\ell k^2/m) = o(\ell)$. We call an assignment α on $[\ell 2^k]$ *bad* if it has more than $\ell/4$ bad elements in its image. Otherwise, we call α good. By Markov's inequality, the probability $\mathcal{A}$ terminates in a bad assignment when executed on $\widetilde{\mu}$ is $o(1)$.

Now consider a good assignment α of weight at most $\ell k/4$. It corresponds to ℓ sub-assignments α_s on $[2^k]$ defined by $\alpha_s(x) = \alpha(s \cdot 2^k + x) \bmod k^{3k}$. Using good, we get that

$$\Pr_{f\sim\widetilde{\mu}}[f \rightharpoonup \alpha] = \Pr_{g\sim\widetilde{\nu}^{t,j}}[g \rightharpoonup \alpha] \tag{5}$$

if the sub-assignment α_t is good and does not cut j. Using that there are at most $\ell/4$ bad α_s and cut, we have that there are at least $\ell k/2$ pairs (t, j) satisfying (5). Hence,

$$\Pr_{f\sim\widetilde{\mu}}[f \rightharpoonup \alpha] \leq 2 \cdot \Pr_{g\sim\nu}[g \rightharpoonup \alpha].$$

Now we finish the proof as in Section 3. ◀

▶ **Theorem 8.** *Assume $\varepsilon n \geq 2$. Then, there exists a non-adaptive 1-sided algorithm that tests a function $f\colon [n] \to [r]$ for monotonicity using $O\big(\frac{\log(\varepsilon n)}{\varepsilon}\big)$ queries.*

Combined with the result of Theorem 7, we get that complexity of this problem is $\Theta\big(\frac{\log(\varepsilon n)}{\varepsilon}\big)$ if $\varepsilon n \geq 2$, and $\Theta(n)$ otherwise.

Proof. The algorithm is inspired by that of Ergün et al. [12]. The main difference is that the i in the loop in line 1c only goes to $\log(\varepsilon n)$ instead of $\log n$.

1. Repeat $\Theta(1/\varepsilon)$ times:
   a. Choose $x \in [n]$ uniformly at random.
   b. Query $f(x)$.
   c. For $i = 0, \ldots, \lceil \log(\varepsilon n) \rceil$:
      - Let w be the largest multiple of 2^i strictly smaller than x, and y be the smallest multiple of 2^i strictly larger than x. (If any of them is outside $[n]$, do not use it on the next steps.)
      - Query $f(w)$ and $f(y)$.
      - If $f(w) > f(x)$ or $f(x) > f(y)$, reject the function f.
2. If no contradiction to monotonicity was found, accept.

Clearly, the query complexity of the algorithm is $O\big(\frac{\log(\varepsilon n)}{\varepsilon}\big)$, it is non-adaptive, and it always accepts a monotone function f. Assume now that f is ε-far from monotone.

▶ **Lemma 9.** *If f is ε-far from monotone, there exists a collection of pairwise disjoint pairs $(x_1, y_1), \ldots, (x_t, y_t)$ for $t = \lfloor \varepsilon n/2 \rfloor$ such that, for all i, $x_i < y_i$, $f(x_i) > f(y_i)$, and $y_i - x_i \leq \varepsilon n$.*

Proof. The pairs can be constructed using the following algorithmic procedure. Start with $S \leftarrow [n]$ and $i \leftarrow 1$. We treat S as a sorted list. While $i \leq t$, choose two neighbouring elements $x_i < y_i$ in S such that $f(x_i) > f(y_i)$, remove x_i and y_i from S, and increment i.

It remains to prove that (a) such a pair (x_i, y_i) will always exist and (b) that $y_i - x_i \leq \varepsilon n$. For (a), observe that $i \leq t$ implies that we have removed strictly less than εn elements from

S so far. Since f is ε-far from monotone, we have that f restricted to S is not monotone (otherwise, it would be possible to extend $f|_S$ to a monotone function on all $[n]$). The existence of the pair (x_i, y_i) is now obvious.

For (b), again, observe that we have removed less than $\varepsilon n - 1$ elements from S so far. The elements x_i and y_i are neighbouring in S, which means they are at distance at most εn in the original list $[n]$. ◀

▶ **Lemma 10.** *Assume x and y satisfy $x < y$, $f(x) > f(y)$ and $y - x \leq \varepsilon n$. Then one element $z \in \{x, y\}$ of these two is such that the algorithm will reject if it chooses z on step 1(a).*

Proof. If $y = x + 1$, the algorithm will reject if it chooses either of x or y. So, assume $y \geq x + 2$.

We claim that there exists an integer $0 \leq i \leq \lceil \log(\varepsilon n) \rceil$ such that there is unique multiple of 2^i strictly between x and y. Indeed, there is at least one for $i = 0$ and at most one for $i = \lceil \log(\varepsilon n) \rceil$. Also, it is not possible that there is more than one for some value of i and zero for $i + 1$.

Let w be this unique multiple of 2^i. If $f(w) < f(x)$, then it will be detected if the algorithm chooses x on step 1(a). If $f(w) \geq f(x)$, then $f(w) > f(y)$, and it will be detected if the algorithm chooses y on step 1(a). ◀

By the previous two lemmata, there exist $\lfloor \varepsilon n/2 \rfloor$ values of x on which the algorithm rejects should it choose any of them on step 1(a). The probability of choosing one of them on one iteration of the loop is $\Omega(\varepsilon)$. The probability to choose any of them on one of the $\Theta(1/\varepsilon)$ iterations of the loop is $\Omega(1)$. ◀

5 The Case of Hypergrids

In this section, we show how our results can be transformed into a lower bound for testing monotonicity on the hypergrid.

▶ **Theorem 11.** *If $\varepsilon \leq 1/2$, the complexity of ε-testing a function $f\colon [n]^d \to [r]$ for monotonicity is*

$$\Omega\left(\min\left\{\frac{\log(\varepsilon n^d)}{\varepsilon}, \frac{\log(\varepsilon r)}{\varepsilon \log\log(\varepsilon r)}\right\}\right).$$

In terms of n, the lower bound can be expressed as $\Omega(\varepsilon^{-1} d \log n - \varepsilon^{-1} \log \varepsilon^{-1})$.

Proof. Consider the functions defined in the proof of Theorem 7. Assume that $\ell = 2^a$ for some integer a, and choose k so that $a + k = db$ for some integer b. We consider the domain of the input functions $[\ell 2^k] = [2^{a+k}]$ as $[2^b]^d$, where we break the binary representation of $x \in [2^{a+k}]$ into d groups of b bits.

If a function is monotone on $[2^{a+k}]$, it is still monotone when considered on $[2^b]^d$. Also, the analysis of the algorithm does not involve the order relation defined on the domain of the input functions. The only thing that might go wrong is that the functions in the support of $\widetilde{\nu}$ are no longer ε-far from monotone when considered on $[2^b]^d$. But this does not happen. Indeed, in the proof of non-monotone, the monotonicity-violating pairs (x, y) differ in exactly one bit. So if $x < y$ in $[2^{a+k}]$, this is still true in $[2^b]^d$, hence the proof of non-monotone carries over. Thus, all the functions in the support of $\widetilde{\nu}$ are ε-far from monotone. The statement of the theorem now follows from the proof of Theorem 7. ◀

References

1 Aleksandrs Belovs and Eric Blais. Quantum algorithm for monotonicity testing on the hypercube. *Theory of Computing*, 11(16):403–412, 2015.

2 Aleksandrs Belovs and Eric Blais. A polynomial lower bound for testing monotonicity. In *Proc. of 48th ACM STOC*, pages 1021–1032, 2016.

3 Eric Blais, Joshua Brody, and Kevin Matulef. Property testing lower bounds via communication complexity. *Computational Complexity*, 21(2):311–358, 2012.

4 Eric Blais, Sofya Raskhodnikova, and Grigory Yaroslavtsev. Lower bounds for testing properties of functions over hypergrid domains. In *Proc. of 29th IEEE CCC*, pages 309–320, 2014.

5 Deeparnab Chakrabarty and C. Seshadhri. An optimal lower bound for monotonicity testing over hypergrids. *Theory of Computing*, 10:453–464, 2014.

6 Deeparnab Chakrabarty and Comandur Seshadhri. A $o(n)$ monotonicity tester for Boolean functions over the hypercube. In *Proc. of 45th ACM STOC*, pages 411–418, 2013.

7 Deeparnab Chakrabarty and Comandur Seshadhri. Optimal bounds for monotonicity and Lipschitz testing over hypercubes and hypergrids. In *Proc. of 45th ACM STOC*, pages 419–428, 2013.

8 Xi Chen, Anindya De, Rocco A. Servedio, and Li-Yang Tan. Boolean function monotonicity testing requires (almost) $n^{1/2}$ non-adaptive queries. In *Proc. of 47th ACM STOC*, pages 519–528, 2015.

9 Xi Chen, Rocco A. Servedio, and Li-Yang Tan. New algorithms and lower bounds for monotonicity testing. In *Proc. of 55th IEEE FOCS*, pages 286–295, 2014.

10 Xi Chen, Erik Waingarten, and Jinyu Xie. Beyond Talagrand functions: new lower bounds for testing monotonicity and unateness. In *Proc. of 49th ACM STOC*, pages 523–536, 2017.

11 Yevgeniy Dodis, Oded Goldreich, Eric Lehman, Sofya Raskhodnikova, Dana Ron, and Alex Samorodnitsky. Improved testing algorithms for monotonicity. In *Proc. of 3rd*, pages 97–108. Springer, 1999.

12 Funda" Ergün, Sampath" "Kannan, S Ravi" "Kumar, Ronitt" "Rubinfeld, and Mahesh "Viswanathan. Spot-checkers. *Journal of Computer and System Sciences*, 60(3):717–751, 2000.

13 Eldar Fischer. On the strength of comparisons in property testing. *Information and Computation*, 189(1):107–116, 2004.

14 Eldar Fischer, Eric Lehman, Ilan Newman, Sofya Raskhodnikova, Ronitt Rubinfeld, and Alex Samorodnitsky. Monotonicity testing over general poset domains. In *Proc. of 34th ACM STOC*, pages 474–483, 2002.

15 Oded Goldreich, Shafi Goldwasser, Eric Lehman, Dana Ron, and Alex Samorodnitsky. Testing monotonicity. *Combinatorica*, 20(3):301–337, 2000.

16 Oded Goldreich, Shari Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45(4):653–750, 1998.

17 Subhash Khot, Dor Minzer, and Muli Safra. On monotonicity testing and Boolean isoperimetric type theorems. In *Proc. of 56th IEEE FOCS*, pages 52–58, 2015.

18 Ramesh Krishnan S Pallavoor, Sofya Raskhodnikova, and Nithin Varma. Parameterized property testing of functions. In *Proc. of 8th ACM ITCS*, volume 67 of *LIPIcs*, page 12. Dagstuhl, 2017.

19 Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM Journal on Computing*, 25(2):252–271, 1996.

Swendsen-Wang Dynamics for General Graphs in the Tree Uniqueness Region

Antonio Blanca
School of Computer Science, Georgia Institute of Technology, Atlanta GA 30332, USA
ablanca3@gatech.edu

Zongchen Chen
School of Mathematics, Georgia Institute of Technology, Atlanta GA 30332, USA
chenzongchen@gatech.edu

Eric Vigoda
School of Computer Science, Georgia Institute of Technology, Atlanta GA 30332, USA
vigoda@gatech.edu

Abstract

The Swendsen-Wang dynamics is a popular algorithm for sampling from the Gibbs distribution for the ferromagnetic Ising model on a graph $G = (V, E)$. The dynamics is a "global" Markov chain which is conjectured to converge to equilibrium in $O(|V|^{1/4})$ steps for any graph G at any (inverse) temperature β. It was recently proved by Guo and Jerrum (2017) that the Swendsen-Wang dynamics has polynomial mixing time on any graph at all temperatures, yet there are few results providing $o(|V|)$ upper bounds on its convergence time.

We prove fast convergence of the Swendsen-Wang dynamics on general graphs in the tree uniqueness region of the ferromagnetic Ising model. In particular, when $\beta < \beta_c(d)$ where $\beta_c(d)$ denotes the uniqueness/non-uniqueness threshold on infinite d-regular trees, we prove that the *relaxation time* (i.e., the inverse spectral gap) of the Swendsen-Wang dynamics is $\Theta(1)$ on any graph of maximum degree $d \geq 3$. Our proof utilizes a version of the Swendsen-Wang dynamics which only updates isolated vertices. We establish that this variant of the Swendsen-Wang dynamics has mixing time $O(\log |V|)$ and relaxation time $\Theta(1)$ on any graph of maximum degree d for all $\beta < \beta_c(d)$. We believe that this Markov chain may be of independent interest, as it is a *monotone* Swendsen-Wang type chain. As part of our proofs, we provide modest extensions of the technology of Mossel and Sly (2013) for analyzing mixing times and of the censoring result of Peres and Winkler (2013). Both of these results are for the Glauber dynamics, and we extend them here to general monotone Markov chains. This class of dynamics includes for example the *heat-bath block dynamics*, for which we obtain new tight mixing time bounds.

2012 ACM Subject Classification Theory of computation → Random walks and Markov chains, Theory of computation → Design and analysis of algorithms

Keywords and phrases Swendsen-Wang dynamics, mixing time, relaxation time, spatial mixing, censoring

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.32

Related Version A full version of the paper is available at `https://arxiv.org/pdf/1806.04602.pdf`.

Funding Research supported in part by NSF grants CCF-1617306 and CCF-1563838.

1 Introduction

For spin systems, sampling from the associated Gibbs distribution is a key computational task with a variety of applications, notably including inference/learning [19] and approximate counting [28, 48]. In the study of spin systems, a model of prominent interest is the Ising model. This is a classical model in statistical physics, which was introduced in the 1920's to study the ferromagnet and its physical phase transition [25, 31]. More recently, the Ising model has found numerous applications in theoretical computer science, computer vision, social network analysis, game theory, biology, discrete probability and many other fields [7, 17, 12, 13, 37].

An instance of the (ferromagnetic) Ising model is given by an undirected graph $G = (V, E)$ on $n = |V|$ vertices and an (inverse) temperature $\beta > 0$. A configuration $\sigma \in \{+, -\}^V$ assigns a spin value ($+$ or $-$) to each vertex $v \in V$. The probability of a configuration σ is proportional to

$$w(\sigma) = \exp\Big(\beta \sum_{\{v,w\}\in E} \sigma(v)\sigma(w)\Big), \tag{1}$$

where $\sigma(v)$ is the spin of v. The associated Gibbs distribution $\mu = \mu_{G,\beta}$ is given by $\mu(\sigma) = w(\sigma)/Z$, where the normalizing factor Z is known as the *partition function*. Since $\beta > 0$ the system is ferromagnetic as neighboring vertices prefer to align their spins.

For general graphs Jerrum and Sinclair [26] presented an FPRAS for the partition function (which yields an efficient sampler); however, its running time is a large polynomial in n. Hence, there is significant interest in obtaining tight bounds on the convergence rate of Markov chains for the Ising model, namely, Markov chains on the space of Ising configurations $\{+, -\}^V$ that converge to Gibbs distribution μ. A standard notion for measuring the speed of convergence to stationarity is the *mixing time*, which is defined as the number of steps until the Markov chain is close to its stationary distribution in total variation distance, starting from the worst possible initial configuration.

A simple, popular Markov chain for sampling from the Gibbs distribution is the Glauber dynamics, commonly referred to as the Gibbs sampler in some communities. This dynamics works by updating a randomly chosen vertex in each step in a reversible fashion. Significant progress has been made in understanding the mixing properties of the Glauber dynamics and its connections to the spatial mixing (i.e., decay of correlation) properties of the underlying spin system. In general, in the high-temperature region (small β) correlations typically decay exponentially fast, and one expects the Glauber dynamics to converge quickly to stationarity. For example, for the special case of the integer lattice $\mathbb{Z}^2$, in the high-temperature region it is well known that the Glauber dynamics has mixing time $\Theta(n \log n)$ [35, 6, 10]. For general graphs, Mossel and Sly [38] proved that the Glauber dynamics mixes in $O(n \log n)$ steps on any graph of maximum degree d in the tree uniqueness region. Tree uniqueness is defined as follows: let T_h denote a (finite) complete tree of height h (by complete we mean all internal vertices have degree d). Fix the leaves to be all $+$ spins, consider the resulting conditional Gibbs distribution on the internal vertices, and let p_h^+ denote the probability the root is assigned spin $+$ in this conditional distribution; similarly, let p_h^- denote the corresponding marginal probability with the leaves fixed to spin $-$. When $\beta < \beta_c(d)$, where $\beta_c(d)$ is such that

$$(d-1)\tanh\beta_c(d) = 1, \tag{2}$$

then $p_\infty^+ = p_\infty^-$ and we say *tree uniqueness* holds since there is a unique Gibbs measure on the infinite d-regular tree [41]. In the same setting, building upon the approach of Weitz [52]

for the hard-core model, Li, Lu and Yin [33] provide an FPTAS for the partition function, but the running time is a large polynomial in n.

In practice, it is appealing to utilize non-local (or global) chains which possibly update $\Omega(n)$ vertices in a step; these chains are more popular due to their presumed speed-up and for their ability to be naturally parallelized [30].

A notable example for the ferromagnetic Ising model is the Swendsen-Wang (SW) dynamics [49] which utilizes the random-cluster representation to derive an elegant Markov chain in which every vertex can change its spin in every step. The SW dynamics works in the following manner. From the current spin configuration $\sigma_t \in \{+,-\}^V$:

1. Consider the set of agreeing edges $E(\sigma_t) = \{(v,w) \in E : \sigma_t(v) = \sigma_t(w)\}$;
2. Independently for each edge $e \in E(\sigma_t)$, "percolate" by deleting e with probability $\exp(-2\beta)$ and keeping e with probability $1 - \exp(-2\beta)$; this yields $F_t \subseteq E(\sigma_t)$;
3. For each connected component C in the subgraph (V, F_t), choose a spin s_C uniformly at random from $\{+,-\}$, and then assign spin s_C to all vertices in C, yielding $\sigma_{t+1} \in \{+,-\}^V$.

The proof that the stationary distribution of the SW dynamics is the Gibbs distribution is non-trivial; see [11] for an elegant proof. The SW dynamics is also well-defined for the *ferromagnetic Potts model*, a natural generalization of the Ising model that allows vertices to be assigned q different spins.

The SW dynamics for the Ising model is quite appealing as it is conjectured to mix quickly at all temperatures.

Its behavior for the Potts model (which corresponds to $q > 2$ spins) is more subtle, as there are multiple examples of classes of graphs where the SW dynamics is torpidly mixing; i.e., mixing time is exponential in the number of vertices of the graph; see, e.g., [20, 15, 3, 18, 4, 5].

Despite the popularity [51, 42, 43] and rich mathematical structure [21] of the SW dynamics there are few results with tight bounds on its speed of convergence to equilibrium. In fact, there are few results proving the SW dynamics is faster than the Glauber dynamics (or the edge dynamics analog in the random-cluster representation). Most results derive as a consequence of analyses of these local dynamics. Recently, Guo and Jerrum [22] established that the mixing time of the SW dynamics on *any* graph and at any temperature is $O(|V|^{10})$.

This bound, however, is far from the conjectured universal upper bound of $O(|V|^{1/4})$ [39], and once again their result derives from a bound on a local chain (the edge dynamics in the random-cluster representation).

In the special case of the *mean-field* Ising model, which corresponds to the underlying graph G being the complete graph on n vertices, Long, Nachmias, Ning and Peres [34] provided a tight analysis of the mixing time of the SW dynamics. They prove that the mixing time of the mean-field SW dynamics is $\Theta(|V|^{1/4})$; this is expected to be the worst case and thus yields the aforementioned conjecture [39].

Another relevant case for which the speed of convergence is known is the two-dimensional integer lattice $\mathbb{Z}^2$ (more precisely, finite subsections of it). Blanca, Caputo, Sinclair and Vigoda [1] recently established that the *relaxation time* of the SW dynamics is $\Theta(1)$ in the high-temperature region. The relaxation time measures the speed of convergence to μ when the initial configuration is reasonably close to this distribution (a so-called "warm start") [27, 29]. More formally, the relaxation time is equal to the inverse spectral gap of the transition matrix of the chain and is another well-studied notion of rate of convergence [32]. This result [1] applied a well-established proof approach [35, 10] which utilizes that $\mathbb{Z}^2$ is an amenable graph. Our goal in this paper is to establish results for general graphs of bounded degree.

Our inspiration is the result of Mossel and Sly [38] who proved $O(n \log n)$ mixing time of the Glauber dynamics for every graph of maximum degree d. When $\beta < \beta_c(d)$, in addition to uniqueness on the infinite d-regular tree, the ferromagnetic Ising model is also known to exhibit several key spatial mixing properties. For instance, Mossel and Sly [38] showed that when $\beta < \beta_c(d)$ a rather strong form of spatial mixing holds on graphs of maximum degree d; see Definition 9 and Lemma 10 in Section 3. Using this, together with the censoring result of Peres and Winkler [40] for the Glauber dynamics, they establish optimal bounds for the mixing and relaxation times of the Glauber dynamics. At a high-level, the censoring result [40] says that extra updates by the Markov chain do not slow it down, and hence one can ignore transitions outside a local region of interest in the analysis of mixing times.

A Markov chain is *monotone* if it preserves the natural partial order on states; see Section 2 for a detailed definition. We generalize the proof approach of Mossel and Sly to apply to general (non-local) monotone Markov chains. This allows us to analyze a monotone variant of the SW dynamics, and a direct comparison of these two chains yields a new bound for the relaxation time of the SW dynamics.

▶ **Theorem 1.** *Let G be an arbitrary n-vertex graph of maximum degree d. If $\beta < \beta_c(d)$, then the relaxation time of the Swendsen-Wang dynamics is $\Theta(1)$.*

This tight bound for the relaxation time is a substantial improvement over the best previously known $O(n)$ bound which follows from Ullrich's comparison theorem [50] combined with Mossel and Sly's result [38] for the Glauber dynamics. We note that in Theorem 1, d is assumed to be a constant independent of n and thus the result holds for arbitrary graphs of *bounded* degree. We also mention that while spatial mixing properties are known to imply optimal mixing of local dynamics, only recently the effects of these properties on the rate of convergence of non-local dynamics have started to be investigated [1]. In general, spatial mixing properties have proved to have a number of powerful algorithmic applications in the design of efficient approximation algorithms for the partition function using the associated self-avoiding walk trees (see, e.g., [52, 45, 33, 16, 44, 46, 47]).

There are three key components in our proof approach. First, we generalize the recursive/inductive argument of Mossel and Sly [38] from the Glauber dynamics to general (non-local) monotone dynamics. Since this approach relies crucially on the censoring result of Peres and Winkler [40] which only applies to the Glauber dynamics, we also need to establish a modest extension of the censoring result. For this, we use the framework of Fill and Kahn [14]. Finally, we require a monotone Markov chain that can be analyzed with these new tools and which is naturally comparable to the SW dynamics. To this end we utilize the *Isolated-vertex dynamics* which was previously used in [1].

The Isolated-vertex dynamics operates in the same manner as the SW dynamics, except in step 3 only components of size 1 choose a new random spin (other components keep the same spin as in σ_t). We prove that the Isolated-vertex dynamics is *monotone*. Combining these new tools we obtain the following result.

▶ **Theorem 2.** *Let G be an arbitrary n-vertex graph of maximum degree d. If $\beta < \beta_c(d)$, then the mixing time of the Isolated-vertex dynamics is $O(\log n)$, and its relaxation time is $\Theta(1)$.*

Our result for censoring may be of independent interest, as it applies to a fairly general class of non-local monotone Markov chains. Indeed, combined with our generalization of Mossel and Sly's results [38], it gives a general method for analyzing monotone Markov chains.

As the first application of this technology, we are able to establish tight bounds for the mixing and relaxation times of the *block dynamics*. Let $\{B_1, ..., B_r\}$ be a collection of sets

(or blocks) such that $B_i \subseteq V$ and $V = \cup_i B_i$. The *heat-bath block dynamics* with blocks $\{B_1, ..., B_r\}$ is a Markov chain that in each step picks a block B_i uniformly at random and updates the configuration in B_i with a new configuration distributed according to the conditional measure in B_i given the configuration in $V \setminus B_i$.

▶ **Theorem 3.** *Let G be an arbitrary n-vertex graph of maximum degree d and let $\{B_1, \dots, B_r\}$ be an arbitrary collection of blocks such that $V = \cup_{i=1}^r B_i$. If $\beta < \beta_c(d)$, then the mixing time of the block dynamics with blocks $\{B_1, \dots, B_r\}$ is $O(r \log n)$, and its relaxation time is $O(r)$.*

We observe that there are no restrictions on the geometry of the blocks B_i in the theorem other than $V = \cup_i B_i$. These optimal bounds were only known before for certain specific collections of blocks.

As a second application of our technology, we consider another monotone variant of the SW dynamics, which we call the *Monotone SW dynamics*. This chain proceeds exactly like the SW dynamics, except that in step 3 each connected component C is assigned a new random spin only with probability $1/2^{|C|-1}$ and is not updated otherwise. We derive the following bounds.

▶ **Theorem 4.** *Let G be an arbitrary n-vertex graph of maximum degree d. If $\beta < \beta_c(d)$, then the mixing time of the Monotone SW dynamics is $O(\log n)$, and its relaxation time is $\Theta(1)$.*

The remainder of the paper is structured as follows. Section 2 contains some basic definitions and facts used throughout the paper. In Section 3 we study the Isolated-vertex dynamics and establish Theorem 2. Theorem 1 for the SW dynamics will follow as an easy corollary of these results. In Section 3 we also state our generalization of Mossel and Sly's approach [38] for non-local dynamics (Theorem 11) and our censoring result (Theorem 7). The proofs of these theorems are included in Appendix A and B, respectively. Finally, the proofs of Theorems 3 and 4 are provided in the full version [2].

2 Background

In this section we provide a number of standard definitions that we will refer to in our proofs. For more details see the book [32].

Ferromagnetic Ising model. Given a graph $G = (V, E)$ and a real number $\beta > 0$, the ferromagnetic Ising model on G consists of the probability distribution over $\Omega_G = \{+, -\}^V$ given by

$$\mu_{G,\beta}(\sigma) = \frac{1}{Z(G,\beta)} \exp\left[\beta \sum\nolimits_{\{u,v\}\in E} \sigma(u)\sigma(v)\right], \tag{3}$$

where $\sigma \in \Omega_G$ and

$$Z(G,\beta) = \sum\nolimits_{\sigma\in\Omega_G} \exp\left[\beta \sum\nolimits_{\{u,v\}\in E} \sigma(u)\sigma(v)\right]$$

is called the *partition function*.

Mixing and relaxation times. Let P be the transition matrix of an ergodic (i.e., irreducible and aperiodic) Markov chain over Ω_G with stationary distribution $\mu = \mu_{G,\beta}$. Let $P^t(X_0, \cdot)$ denote the distribution of the chain after t steps starting from $X_0 \in \Omega_G$, and let

$$T_{\text{mix}}(P, \varepsilon) = \max_{X_0\in\Omega} \min\left\{t \geq 0 : \|P^t(X_0, \cdot) - \mu(\cdot)\|_{\text{TV}} \leq \varepsilon\right\}.$$

The *mixing time* of P is defined as $T_{\text{mix}}(P) = T_{\text{mix}}(P, 1/4)$.

If P is reversible with respect to (w.r.t.) μ, the spectrum of P is real. Let $1 = \lambda_1 > \lambda_2 \geq ... \geq \lambda_{|\Omega|} \geq -1$ denote its eigenvalues. The *absolute spectral gap* of P is defined by $\lambda(P) = 1 - \lambda^*$, where $\lambda^* = \max\{|\lambda_2|, |\lambda_{|\Omega|}|\}$. $T_{\text{rel}}(P) = \lambda(P)^{-1}$ is called the *relaxation time* of P, and is another well-studied notion of rate of convergence to μ [27, 29].

Couplings and grand couplings. A *(one step) coupling* of a Markov chain $\mathcal{M}$ over Ω_G specifies, for every pair of states $(X_t, Y_t) \in \Omega_G \times \Omega_G$, a probability distribution over (X_{t+1}, Y_{t+1}) such that the processes $\{X_t\}$ and $\{Y_t\}$, viewed in isolation, are faithful copies of $\mathcal{M}$, and if $X_t = Y_t$ then $X_{t+1} = Y_{t+1}$. Let $\{X_t^\sigma\}_{t\geq 0}$ denote an instance of $\mathcal{M}$ started from $\sigma \in \Omega_G$. A *grand coupling* of $\mathcal{M}$ is a simultaneous coupling of $\{X_t^\sigma\}_{t\geq 0}$ for all $\sigma \in \Omega_G$.

Monotonicity. For two configurations $\sigma, \tau \in \Omega_G$, we say $\sigma \geq \tau$ if $\sigma(v) \geq \tau(v)$ for all $v \in V$ (assuming "+" > "−"). This induces a partial order on Ω_G. The ferromagnetic Ising model is *monotone* w.r.t. this partial order, since for every $B \subseteq V$ and every pair of configurations τ_1, τ_2 on B such that $\tau_1 \geq \tau_2$ we have $\mu(\cdot \mid \tau_1) \succeq \mu(\cdot \mid \tau_2)$, where $\succeq$ denotes stochastic domination. (For two distributions ν_1, ν_2 on Ω_G, we say that ν_1 stochastically dominates ν_2 if for any increasing function $f \in \mathbb{R}^{|\Omega_G|}$ we have $\sum_{\sigma\in\Omega_G} \nu_1(\sigma)f(\sigma) \geq \sum_{\sigma\in\Omega_G} \nu_2(\sigma)f(\sigma)$, where a vector or function $f \in \mathbb{R}^{|\Omega_G|}$ is increasing if $f(\sigma) \geq f(\tau)$ for all $\sigma \geq \tau$.)

Suppose $\mathcal{M}$ is an ergodic Markov chain over Ω_G with stationary distribution μ and transition matrix P. A coupling of two instances $\{X_t\}$, $\{Y_t\}$ of $\mathcal{M}$ is a *monotone coupling* if $X_{t+1} \geq Y_{t+1}$ whenever $X_t \geq Y_t$. We say that $\mathcal{M}$ is a *monotone Markov chain* and P is a *monotone transition matrix* if $\mathcal{M}$ has a monotone grand coupling.

Comparison inequalities. The *Dirichlet form* of a Markov chain with transition matrix P reversible w.r.t. μ is defined for any $f, g \in \mathbb{R}^{|\Omega_G|}$ as

$$\mathcal{E}_P(f, g) = \langle f, (I - P)g\rangle_\mu = \frac{1}{2} \sum_{\sigma,\tau\in\Omega_G} \mu(\sigma)P(\sigma, \tau)(f(\sigma) - f(\tau))(g(\sigma) - g(\tau)),$$

where $\langle f, g\rangle_\mu = \sum_{\sigma\in\Omega_G} \mu(\sigma)f(\sigma)g(\sigma)$ for all $f, g \in \mathbb{R}^{|\Omega_G|}$.

If P and Q are the transition matrices of two monotone Markov chains reversible w.r.t. μ, we say that $P \leq Q$ if $\langle Pf, g\rangle_\mu \leq \langle Qf, g\rangle_\mu$ for every increasing and positive $f, g \in \mathbb{R}^{|\Omega_G|}$. Note that $P \leq Q$ is equivalent to $\mathcal{E}_P(f, g) \geq \mathcal{E}_Q(f, g)$ for every increasing and positive $f, g \in \mathbb{R}^{|\Omega_G|}$.

3 Isolated-vertex dynamics

In this section we consider a variant of the SW dynamics known as the *Isolated-vertex dynamics* which was first introduced in [1]. We shall use this dynamics to introduce a general framework for analyzing monotone Markov chains for the Ising model and to derive our bounds for the SW dynamics. Specifically, we will prove Theorems 1 and 2 from the introduction.

Throughout the section, let $G = (V, E)$ be an arbitrary n-vertex graph of maximum degree d, $\mu = \mu_{G,\beta}$ and $\Omega = \Omega_G$. Given an Ising model configuration $\sigma_t \in \Omega$, one step of the Isolated-vertex dynamics is given by:

1. Consider the set of agreeing edges $E(\sigma_t) = \{(v, w) \in E : \sigma_t(v) = \sigma_t(w)\}$;
2. Independently for each edge $e \in E(\sigma_t)$, delete e with probability $\exp(-2\beta)$ and keep e with probability $1 - \exp(-2\beta)$; this yields $F_t \subseteq E(\sigma_t)$;

3. For each *isolated* vertex v in the subgraph (V, F_t) (i.e., those vertices with no incident edges in F_t), choose a spin uniformly at random from $\{+, -\}$ and assign it to v to obtain σ_{t+1}; all other (non-isolated) vertices keep the same spin as in σ_t.

We use $\mathcal{IV}$ to denote the transition matrix of this chain. The reversibility of $\mathcal{IV}$ with respect to μ was established in [1]. Observe also that in step 3, only *isolated vertices* are updated with new random spins, whereas in the SW dynamics all connected components are assigned new random spins. It is thus intuitive that the SW dynamics converges faster to stationarity than the Isolated-vertex dynamics. This intuition was partially captured in [1], where it was proved that

$$T_{\text{rel}}(\mathcal{SW}) \leq T_{\text{rel}}(\mathcal{IV}). \tag{4}$$

The Isolated-vertex dynamics exhibits various properties that vastly simplify its analysis. These properties allow us to deduce, for example, strong bounds for both its relaxation and mixing times. Specifically, we show (see Theorem 2) that when $\beta < \beta_c(d)$, $T_{\text{mix}}(\mathcal{IV}) = O(\log n)$ and $T_{\text{rel}}(\mathcal{IV}) = \Theta(1)$; see also (2) for the definition of $\beta_c(d)$. Theorem 1 from the introduction then follows from (4).

A comparison inequality like (4) but for mixing times is not known, so Theorem 2 does not yield a $O(\log n)$ bound for the mixing time of the SW dynamics as one might hope. Direct comparison inequalities for mixing times are rare, since almost all known techniques involve the comparison of Dirichlet forms, and there are inherent penalties in using such inequalities to derive mixing times bounds.

The first key property of the Isolated-vertex dynamics is that, unlike the SW dynamics, this Markov chain is *monotone*. Monotonicity is known to play a key role in relating spatial mixing (i.e., decay of correlation) properties to fast convergence of the Glauber dynamics. For instance, for spin systems in lattice graphs, sophisticated functional analytic techniques are required to establish the equivalence between a spatial mixing property known as *strong spatial mixing* and optimal mixing of the Glauber dynamics [35, 36, 37]. For monotone spin systems such as the Ising model a simpler combinatorial argument yields the same sharp result [10]. This combinatorial argument is in fact more robust, since it can be used to analyze a larger class of Markov chains, including for example the systematic scan dynamics [1].

▶ **Lemma 5.** *For all graphs G and all $\beta > 0$, the Isolated-vertex dynamics for the Ising model is monotone.*

The proof of Lemma 5 is given in Section 3.1. The second key property of the Isolated-vertex dynamics concerns whether moves (or partial moves) of the dynamics could be *censored* from the evolution of the chain without possibly speeding up its convergence. Censoring of Markov chains is a well-studied notion [40, 14, 24] that has found important applications [38, 8, 9].

We say that a stochastic $|\Omega| \times |\Omega|$ matrix Q acts on a set $A \subseteq V$ if for all $\sigma, \sigma' \in \Omega$:

$$Q(\sigma, \sigma') \neq 0 \text{ iff } \sigma(V \setminus A) = \sigma'(V \setminus A).$$

Also recall that $P \leq P_A$ if $\langle Pf, g\rangle_\mu \leq \langle P_A f, g\rangle_\mu$ for any pair of increasing positive functions $f, g \in \mathbb{R}^{|\Omega|}$.

▶ **Definition 6.** Let G be an arbitrary graph and let $\beta > 0$. Consider an ergodic and monotone Markov chain for the Ising model on G, reversible w.r.t. $\mu = \mu_{G,\beta}$ with transition matrix P. Let $\{P_A\}_{A \subseteq V}$ be a collection of monotone stochastic matrices reversible w.r.t. μ with the property that P_A acts on A for every $A \subseteq V$. We say that $\{P_A\}_{A \subseteq V}$ is a *censoring* for P if $P \leq P_A$ for all $A \subseteq V$.

As an example, consider the *heat-bath Glauber dynamics* for the Ising model on the graph $G = (V, E)$. Recall that in this Markov chain a vertex $v \in V$ is chosen uniformly at random (u.a.r.) and a new spin is sampled for v from the conditional distribution at v given the configuration on $V \setminus v$. For every $A \subseteq V$, we may take P_A to be the $|\Omega| \times |\Omega|$ transition matrix of the censored heat-bath Glauber dynamics that ignores all moves outside of A. That is, if the randomly chosen vertex $v \in V$ is not in A, then the move is ignored; otherwise the chain proceeds as the standard heat-bath Glauber dynamics.

It is easy to check that P_A is monotone and reversible w.r.t. μ. Moreover, it was established in [40, 14] that $P \leq P_A$ for every $A \subseteq V$, and thus the collection $\{P_A\}_{A \subseteq V}$ is a censoring for the heat-bath Glauber dynamics. This particular censoring has been used to analyze the speed of convergence of the Glauber dynamics in various settings (see [38, 40, 8, 9]), since it can be proved that censored variants of the Glauber dynamics—where moves of P are replaced by moves of P_A—converge more slowly to the stationary distribution [40, 14]. Consequently, it suffices to analyze the speed of convergence of the censored chain, and this could be much simpler for suitably chosen censoring schemes.

Using the machinery from [40, 14], we can show that given a censoring (as defined in Definition 6), the strategy just mentioned for Glauber dynamics can be used for general monotone Markov chains.

▶ **Theorem 7.** *Let G be an arbitrary graph and let $\beta > 0$. Let $\{X_t\}$ be an ergodic monotone Markov chain for the Ising model on G, reversible w.r.t. $\mu = \mu_{G,\beta}$ with transition matrix P. Let $\{P_A\}_{A \subseteq V}$ be a censoring for P and let $\{\hat{X}_t\}$ be a censored version of $\{X_t\}$ that sequentially applies $P_{A_1}, P_{A_2}, P_{A_3} \dots$ where $A_i \subseteq V$. If X_0, Y_0 are both sampled from a distribution ν over Ω such that ν/μ is increasing, then the following hold:*

1. *$X_t \preceq \hat{X}_t$ for all $t \geq 0$;*
2. *Let $\hat{P}^t = P_{A_1} \dots P_{A_t}$. Then, for all $t \geq 0$*

$$\|P^t(X_0, \cdot) - \mu(\cdot)\|_{\text{TV}} \leq \|\hat{P}^t(X_0, \cdot) - \mu(\cdot)\|_{\text{TV}}.$$

If ν/μ is decreasing, then $X_t \succeq \hat{X}_t$ for all $t \geq 0$.

The proof of this theorem is provided in Appendix B.

We define next a specific censoring for the Isolated-vertex dynamics. For $A \subseteq V$, let $\mathcal{IV}_A$ be the transition matrix for the Markov chain that given an Ising model configuration σ_t generates σ_{t+1} as follows:

1. Consider the set of agreeing edges $E(\sigma_t) = \{(v, w) \in E : \sigma_t(v) = \sigma_t(w)\}$;
2. Independently for each edge $e \in E(\sigma_t)$, delete e with probability $\exp(-2\beta)$ and keep e with probability $1 - \exp(-2\beta)$; this yields $F_t \subseteq E(\sigma_t)$;
3. For each isolated vertex v of the subgraph (V, F_t) in the subset A, choose a spin uniformly at random from $\{+, -\}$ and assign it to v to obtain σ_{t+1}; all other vertices keep the same spin as in σ_t.

▶ **Lemma 8.** *The collection of matrices $\{\mathcal{IV}_A\}_{A \subseteq V}$ is a censoring for the Isolated-vertex dynamics.*

The proof of Lemma 8 is provided in Secion 3.2. To establish Theorem 2 we show that a strong form of spatial mixing, which is known to hold for all $\beta < \beta_c(d)$ [38], implies the desired mixing and relaxation times bounds for the Isolated-vertex dynamics. We define this notion of spatial mixing next.

For $v \in V$ and $R \in \mathbb{N}$, let $B(v, R) = \{u \in V : \text{dist}(u, v) \leq R\}$ denote the ball of radius R around v, where $\text{dist}(\cdot, \cdot)$ denotes graph distance. Also, let $S(v, R) = B(v, R + 1) \backslash B(v, R)$

be the external boundary of $B(v,R)$. For any $A \subseteq V$, let $\Omega_A = \{+,-\}^A$ be the set of all configurations on A; hence $\Omega = \Omega_G = \Omega_V$. For $v \in V$, $u \in S(v,R)$ and $\tau \in \Omega_{S(v,R)}$, let τ_u^+ (resp., τ_u^-) be the configuration obtained from τ by changing the spin of u to $+$ (resp., to $-$) and define

$$a_u = \sup_{\tau \in \Omega_{S(v,R)}} \left| \mu\left(v = + \mid S(v,R) = \tau_u^+\right) - \mu\left(v = + \mid S(v,R) = \tau_u^-\right) \right|, \tag{5}$$

where "$v = +$" represents the event that the spin of v is $+$ and "$S(v,R) = \tau_u^+$" (resp., "$S(v,R) = \tau_u^-$") stands for the event that $S(v,R)$ has configuration τ_u^+ (resp., τ_u^-).

▶ **Definition 9.** We say that *Aggregate Strong Spatial Mixing (ASSM)* holds for $R \in \mathbb{N}$, if for all $v \in V$

$$\sum_{u \in S(v,R)} a_u \leq \frac{1}{4}.$$

▶ **Lemma 10** (Lemma 3, [38]). *For all graphs G of maximum degree d and all $\beta < \beta_c(d)$, there exists an integer $R = R(\beta, d) \in \mathbb{N}$ such that ASSM holds for R.*

Theorem 2 is then a direct corollary of the following more general theorem. The proof of this general theorem, which is provided in Appendix A, follows closely the approach in [38] for the case of the Glauber dynamics, but key additional considerations are required to establish such result for general (non-local) monotone Markov chains. The main new innovation in our proof is the use of the more general Theorem 7, instead of the standard censoring result in [40].

▶ **Theorem 11.** *Let $\beta > 0$ and G be an arbitrary n-vertex graph of maximum degree d where d is a constant independent of n. Consider an ergodic monotone Markov chain for the Ising model on G, reversible w.r.t. $\mu = \mu_{G,\beta}$ with transition matrix P. Suppose $\{P_A\}_{A \subseteq V}$ is a censoring for P. If ASSM holds for a constant $R > 0$, and for any $v \in V$ and any starting configuration $\sigma \in \Omega$*

$$T_{\text{mix}}(P_{B(v,R)}) \leq T, \tag{6}$$

then $T_{\text{mix}}(P) = O(T \log n)$ and $T_{\text{rel}}(P) = O(T)$.

We note that $T_{\text{mix}}(P_{B(v,R)})$ denotes the mixing time from the worst possible starting configuration, both in $B(v,R)$ and in $V \setminus B(v,R)$. (Since $P_{B(v,R)}$ only acts in $B(v,R)$, the configuration in $V \setminus B(v,R)$ remains fixed throughout the evolution of the chain and determines its stationary distribution.)

We now use Theorem 11 to establish Theorem 2. Theorem 11 is also used to establish Theorems 3 and 4 from the introduction, concerning the mixing time of the block dynamics and a monotone variant of the SW dynamics; see the full version of this paper [2].

Proof of Theorem 2. By Lemma 5 the Isolated-vertex dynamics is monotone, and by Lemma 8 the collection $\{\mathcal{IV}_A\}_{A \subseteq V}$ is a censoring for $\mathcal{IV}$. Moreover, Lemma 10 implies that there exists a constant R such that ASSM holds. Thus, to apply Theorem 11 all that is needed is a bound for $T_{\text{mix}}(\mathcal{IV}_{B(v,R)})$ for all $v \in V$. For this, we can use a crude coupling argument. Since $|B(v,R)| \leq d^R$, the probability that every vertex in $B(v,R)$ becomes isolated is at least

$$\mathrm{e}^{-2\beta d|B(v,R)|} \geq \mathrm{e}^{-2\beta d^{R+1}}.$$

Starting from two arbitrary configurations in $B(v, R)$, if all vertices become isolated in both configurations, then we can couple them with probability 1. Hence, we can couple two arbitrary configurations in one step with probability at least $\exp(-2\beta d^{R+1})$. Thus, $T_{\text{mix}}(\mathcal{IV}_{B(v,R)}) = \exp(O(\beta d^{R+1})) = O(1)$, and the result then follows from Theorem 11. ◀

Proof of Theorem 1. Follows from Theorem 2 and the fact that $T_{\text{rel}}(\mathcal{SW}) \leq T_{\text{rel}}(\mathcal{IV})$, which was established in Lemma 4.1 from [1]. ◀

3.1 Monotonicity of the Isolated-vertex dynamics

In this section, we show that the Isolated-vertex dynamics is monotone by constructing a monotone grand coupling; see Section 2 for the definition of a grand coupling. In particular, we prove Lemma 5.

Proof of Lemma 5. Let $\{X_t^\sigma\}_{t\geq 0}$ be an instance of the Isolated-vertex dynamics starting from $\sigma \in \Omega$; i.e., $X_0^\sigma = \sigma$. We construct a grand coupling for the Isolated-vertex dynamics as follows. At time t:

1. For every edge $e \in E$, pick a number $r_t(e)$ uniformly at random from $[0, 1]$;
2. For every vertex $v \in V$, choose a uniform random spin $s_t(v)$ from $\{+, -\}$;
3. For every $\sigma \in \Omega$:
   - (i) Obtain $F_t^\sigma \subseteq E$ by including the edge $e = \{u, v\}$ in F_t^σ iff $X_t^\sigma(u) = X_t^\sigma(v)$ and $r_t(e) \leq 1 - \mathrm{e}^{-2\beta}$;
   - (ii) For every $v \in V$, set $X_{t+1}^\sigma(v) = s_t(v)$ if v is an *isolated vertex* in the subgraph (V, F_t^σ); otherwise, set $X_{t+1}^\sigma(v) = X_t^\sigma(v)$.

This is clearly a valid grand coupling for the Isolated-vertex dynamics. We show next that it is also monotone.

Suppose $X_t^\sigma \geq X_t^\tau$. We need to show that $X_{t+1}^\sigma \geq X_{t+1}^\tau$ after one step of the grand coupling. Let $v \in V$. If v is not isolated in either (V, F_t^σ) or (V, F_t^τ), then the spin of v remains unchanged in both X_{t+1}^σ and X_{t+1}^τ, and $X_{t+1}^\sigma(v) = X_t^\sigma(v) \geq X_t^\tau(v) = X_{t+1}^\tau(v)$. On the other hand, if v is isolated in both (V, F_t^σ) and (V, F_t^σ), then the spin of v is set to $s_t(v)$ in both instances of the chain; hence, $X_{t+1}^\sigma(v) = s_t(v) = X_{t+1}^\tau(v)$.

Suppose next that v is isolated in (V, F_t^σ) but not in (V, F_t^τ). Then, $X_{t+1}^\sigma(v) = s_t(v)$ and $X_{t+1}^\tau(v) = X_t^\tau(v)$. The only possibility that would violate $X_{t+1}^\sigma(v) \geq X_{t+1}^\tau(v)$ is that $X_{t+1}^\sigma(v) = -, X_t^\sigma(v) = +$ and $X_{t+1}^\tau(v) = X_t^\tau(v) = +$. If this is the case, then $X_t^\sigma(v) = X_t^\tau(v) = +$. Moreover, since $X_t^\sigma \geq X_t^\tau$, all neighbors of v assigned "+" in X_t^τ are also "+" in X_t^σ; thus if v is isolated in (V, F_t^σ) then v is also isolated in (V, F_t^τ). This leads to a contradiction, and so $X_{t+1}^\sigma(v) \geq X_{t+1}^\tau(v)$. The case in which v is isolated in (V, F_t^τ) but not in (V, F_t^σ) follows from an analogous argument. ◀

We can use the same grand coupling to show that $\mathcal{IV}_A$ is also monotone for all $A \subseteq V$. The only required modification in the construction is that if $v \in V \setminus A$, then the spin of v is not updated in either copy. This gives the following corollary.

▶ **Corollary 12.** *$\mathcal{IV}_A$ is monotone for all $A \subseteq V$.*

3.2 Censoring for the Isolated-vertex dynamics

In this section we show that the collection $\{\mathcal{IV}_A\}_{A\subseteq V}$ is a censoring for $\mathcal{IV}$. Specifically, we prove Lemma 8.

Proof of Lemma 8. For all $A \subseteq V$, we need to establish that $\mathcal{IV}_A$ is reversible w.r.t. $\mu = \mu_{G,\beta}$, monotone and that $\mathcal{IV} \leq \mathcal{IV}_A$. Monotonicity follows from Corollary 12. To establish the other two facts we use an alternative representation of the matrices $\mathcal{IV}$ and $\mathcal{IV}_A$ that was already used in [1] and is inspired by the methods in [50].

Let $\Omega_{\text{J}} = 2^E \times \Omega$ be the *joint* configuration space, where configurations consist of a spin assignment to the vertices together with a subset of the edges of G. The joint Edwards-Sokal measure ν on Ω_{J} is given by

$$\nu(F,\sigma) = \frac{1}{Z_{\text{J}}} p^{|F|}(1-p)^{|E\setminus F|}\, \mathbb{1}(F \subseteq E(\sigma)), \tag{7}$$

where $p = 1 - \mathrm{e}^{-2\beta}$, $F \subseteq E$, $\sigma \in \Omega$, $E(\sigma) = \{\{u,v\} \in E : \sigma(u) = \sigma(v)\}$, and Z_{J} is the partition function [11].

Let T be the $|\Omega| \times |\Omega_{\text{J}}|$ matrix given by:

$$T(\sigma,(F,\tau)) = \mathbb{1}(\sigma = \tau)\, \mathbb{1}(F \subseteq E(\sigma)) p^{|F|}(1-p)^{|E(\sigma)\setminus F|}, \tag{8}$$

where $\sigma \in \Omega$ and $(F,\tau) \in \Omega_{\text{J}}$. The matrix T corresponds to adding each edge $\{u,v\} \in E$ with $\sigma(u) = \sigma(v)$ independently with probability p, as in step 1 of the Isolated-vertex dynamics. Let $L_2(\nu)$ and $L_2(\mu)$ denote the Hilbert spaces $(\mathbb{R}^{|\Omega_{\text{J}}|}, \langle\cdot,\cdot\rangle_\nu)$ and $(\mathbb{R}^{|\Omega|}, \langle\cdot,\cdot\rangle_\mu)$ respectively. The matrix T defines an operator from $L_2(\nu)$ to $L_2(\mu)$ via vector-matrix multiplication. Specifically, for any $f \in \mathbb{R}^{|\Omega_{\text{J}}|}$ and $\sigma \in \Omega$

$$Tf(\sigma) = \sum_{(F,\tau)\in\Omega_{\text{J}}} T(\sigma,(F,\tau)) f(F,\tau).$$

It is easy to check that the adjoint operator $T^* : L_2(\mu) \to L_2(\nu)$ of T is given by the $|\Omega_{\text{J}}| \times |\Omega|$ matrix

$$T^*((F,\tau),\sigma) = \mathbb{1}(\tau = \sigma), \tag{9}$$

with $(F,\tau) \in \Omega_{\text{J}}$ and $\sigma \in \Omega$. Finally, for $A \subseteq V$, $F_1, F_2 \subseteq E$ and $\sigma,\tau \in \Omega$ let

$$\begin{aligned} Q_A((F_1,\sigma),(F_2,\tau)) = {} & \mathbb{1}(F_1 = F_2)\, \mathbb{1}(F_1 \subseteq E(\sigma) \cap E(\tau)) \\ & \mathbb{1}(\sigma(\mathcal{I}_A^c(F_1)) = \tau(\mathcal{I}_A^c(F_1))) \cdot 2^{-|\mathcal{I}_A(F_1)|} \end{aligned}$$

where $\mathcal{I}_A(F_1)$ is the set of isolated vertices of (V,F_1) in A and $\mathcal{I}_A^c(F_1) = V \setminus \mathcal{I}_A(F_1)$, and similarly for F_2. For ease of notation we set $Q = Q_V$. It follows straightforwardly from the definition of these matrices that $\mathcal{IV} = TQT^*$ and $\mathcal{IV}_A = TQ_AT^*$ for all $A \subseteq V$. It is also easy to verify that $Q = Q^2 = Q^*$, $Q_A = Q_A^2 = Q_A^*$ and that $Q = Q_AQQ_A$; see [1].

The reversibility of $\mathcal{IV}_A$ w.r.t. μ follows from the fact that $\mathcal{IV}_A^* = (TQ_AT^*)^* = TQ_AT^* = \mathcal{IV}_A$. This implies that $\mathcal{IV}_A$ is self-adjoint and thus reversible w.r.t. μ [32].

To establish that $\mathcal{IV} \leq \mathcal{IV}_A$, it is sufficient to show that for every pair of increasing and positive functions $f_1, f_2 : \mathbb{R}^{|\Omega|} \to \mathbb{R}$ on Ω, we have

$$\langle f_1, \mathcal{IV} f_2\rangle_\mu \leq \langle f_1, \mathcal{IV}_A f_2\rangle_\mu. \tag{10}$$

Now,

$$\langle f_1, \mathcal{IV}_A f_2\rangle_\mu = \langle f_1, TQ_AT^*f_2\rangle_\mu = \langle f_1, TQ_A^2T^*f_2\rangle_\mu = \langle Q_AT^*f_1, Q_AT^*f_2\rangle_\nu = \langle \hat{f}_1, \hat{f}_2\rangle_\nu,$$

where $\hat{f}_1 = Q_AT^*f_1$ and $\hat{f}_2 = Q_AT^*f_2$. Similarly,

$$\langle f_1, \mathcal{IV} f_2\rangle_\mu = \langle f_1, TQ_AQ^2Q_AT^*f_2\rangle_\mu = \langle QQ_AT^*f_1, QQ_AT^*f_2\rangle_\nu = \langle Q\hat{f}_1, Q\hat{f}_2\rangle_\nu.$$

Thus, it is sufficient for us to show that $\langle Q\hat{f}_1, Q\hat{f}_2\rangle_\nu \leq \langle \hat{f}_1, \hat{f}_2\rangle_\nu$.

Consider the partial order on Ω_{J} where $(F,\sigma) \geq (F',\sigma')$ iff $F = F'$ and $\sigma \geq \sigma'$.

▶ **Claim 13.** *Suppose $f : \mathbb{R}^{|\Omega|} \to \mathbb{R}$ is an increasing positive function. Then, $\hat{f} : \mathbb{R}^{|\Omega_{\mathrm{J}}|} \to \mathbb{R}$ where $\hat{f} = Q_A T^* f$ is also increasing and positive.*

Given $\omega \in \Omega_{\mathrm{J}}$, let $\rho_\omega(\cdot) = Q(\omega, \cdot)$; i.e., ρ_ω is the distribution over Ω_{J} after applying Q from ω. We have

$$Q\hat{f}_1(\omega) = \sum_{\omega' \in \Omega_{\mathrm{J}}} Q\big(\omega, \omega'\big)\hat{f}_1(\omega') = \mathbb{E}_{\rho_\omega}[\hat{f}_1].$$

Similarly, we get $Q\hat{f}_2(\omega) = \mathbb{E}_{\rho_\omega}[\hat{f}_2]$.

For a distribution π on a partially ordered set S, we say π is positively correlated if for any increasing functions $f, g \in \mathbb{R}^{|S|}$ we have $\mathbb{E}_\pi[fg] \geq \mathbb{E}_\pi[f]\,\mathbb{E}_\pi[g]$. Since ρ_ω is a product distribution over the isolated vertices in ω, ρ_ω is positively correlated for any $\omega \in \Omega_{\mathrm{J}}$ by Harris inequality (see, e.g., Lemma 22.14 in [32]). By Claim 13, $\hat{f}_1$ and $\hat{f}_2$ are increasing. We then deduce that for any $\omega \in \Omega_{\mathrm{J}}$:

$$Q\hat{f}_1(\omega)\, Q\hat{f}_2(\omega) = \mathbb{E}_{\rho_\omega}[\hat{f}_1]\,\mathbb{E}_{\rho_\omega}[\hat{f}_2] \leq \mathbb{E}_{\rho_\omega}[\hat{f}_1\,\hat{f}_2].$$

Putting all these facts together, we get

$$\begin{aligned}\langle Q\hat{f}_1, Q\hat{f}_2\rangle_\nu &= \sum_{\omega\in\Omega_{\mathrm{J}}} Q\hat{f}_1(\omega)\, Q\hat{f}_2(\omega)\nu(\omega) \leq \sum_{\omega\in\Omega_{\mathrm{J}}} \mathbb{E}_{\rho_\omega}[\hat{f}_1\,\hat{f}_2]\nu(\omega)\\ &= \sum_{\omega,\omega'\in\Omega_{\mathrm{J}}} \hat{f}_1(\omega')\,\hat{f}_2(\omega')\rho_\omega(\omega')\nu(\omega) = \sum_{\omega,\omega'\in\Omega_{\mathrm{J}}} \hat{f}_1(\omega')\,\hat{f}_2(\omega')\rho_{\omega'}(\omega)\nu(\omega')\\ &= \langle \hat{f}_1, \hat{f}_2\rangle_\nu,\end{aligned}$$

where the second to last equality follows from the reversibility of Q w.r.t. ν; namely,

$$\rho_\omega(\omega')\nu(\omega) = Q(\omega, \omega')\nu(\omega) = Q(\omega', \omega)\nu(\omega') = \rho_{\omega'}(\omega)\nu(\omega').$$

This implies that (10) holds for every pair of increasing positive functions, and the theorem follows. ◀

We conclude this section with the proof of Claim 13.

Proof of Claim 13. From the definition of T^* we get $T^* f(F, \sigma) = f(\sigma)$ for any $(F, \sigma) \in \Omega_{\mathrm{J}}$. Let $(F, \sigma), (F, \tau) \in \Omega_{\mathrm{J}}$ be such that $\sigma \geq \tau$. Then,

$$\hat{f}(F, \sigma) = Q_A T^* f(F, \sigma) = \sum_{(F', \sigma') \in \Omega_{\mathrm{J}}} Q_A\big((F, \sigma), (F', \sigma')\big) f(\sigma').$$

Recall that $Q_A\big((F, \sigma), (F', \sigma')\big) > 0$ iff $F = F'$ and σ, σ' differ only in $\mathcal{I}_A(F)$, the set of isolated vertices in A. If this is the case, then

$$Q_A\big((F, \sigma), (F, \sigma')\big) = \frac{1}{2^{|\mathcal{I}_A(F)|}}.$$

For $\xi \in \Omega_{\mathcal{I}_A(F)}$, let σ_ξ denote the configuration obtained from σ by changing the spins of vertices in $\mathcal{I}_A(F)$ to ξ; τ_ξ is defined similarly. (Recall that $\Omega_{\mathcal{I}_A(F)}$ denotes the set of Ising configurations on the set $\mathcal{I}_A(F)$.) Then, $\sigma_\xi \geq \tau_\xi$ for any $\xi \in \Omega_{\mathcal{I}_A(F)}$ and

$$\hat{f}(F, \sigma) = \frac{1}{2^{|\mathcal{I}_A(F)|}} \sum_{\xi \in \Omega_{\mathcal{I}_A(F)}} f(\sigma_\xi) \geq \frac{1}{2^{|\mathcal{I}_A(F)|}} \sum_{\xi \in \Omega_{\mathcal{I}_A(F)}} f(\tau_\xi) = \hat{f}(F, \tau).$$

This shows that $\hat{f}$ is increasing. ◀

References

1. A. Blanca, P. Caputo, A. Sinclair, and E. Vigoda. Spatial Mixing and Non-local Markov chains. In *Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1965–1980, 2018.
2. A. Blanca, Z. Chen, and E. Vigoda. Swendsen-Wang Dynamics for General Graphs in the Tree Uniqueness Region. *ArXiv preprint arXiv:1806.04602*, 2018.
3. A. Blanca and A. Sinclair. Dynamics for the mean-field random-cluster model. In *Proceedings of the 19th International Workshop on Randomization and Computation (RANDOM)*, pages 528–543, 2015.
4. C. Borgs, J. Chayes, and P. Tetali. Tight bounds for mixing of the Swendsen-Wang algorithm at the Potts transition point. *Probability Theory and Related Fields*, 152(3-4):509–557, 2012.
5. C. Borgs, A. M. Frieze, J. H. Kim, P. Tetali, E. Vigoda, and V. Vu. Torpid mixing of some Monte Carlo Markov chain algorithms in statistical physics. In *Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 218–229, 1999.
6. F. Cesi. Quasi–factorization of the entropy and logarithmic Sobolev inequalities for Gibbs random fields. *Probability Theory and Related Fields*, 120(4):569–584, 2001.
7. C. Daskalakis, E. Mossel, and S. Roch. Evolutionary trees and the Ising model on the Bethe lattice: a proof of Steel's conjecture. *Probability Theory and Related Fields*, 149(1-2):149–189, 2011.
8. J. Ding, E. Lubetzky, and Y. Peres. Mixing time of critical Ising model on trees is polynomial in the height. *Communications in Mathematical Physics*, 295(1):161–207, 2010.
9. J. Ding and Y. Peres. Mixing time for the Ising model: a uniform lower bound for all graphs. *Annales de l'Institut Henri Poincaré, Probabilités et Statistiques*, 47(4):1020–1028, 2011.
10. M. Dyer, A. Sinclair, E. Vigoda, and D. Weitz. Mixing in time and space for lattice spin systems: A combinatorial view. *Random Structures & Algorithms*, 24(4):461–479, 2004.
11. R. G. Edwards and A. D. Sokal. Generalization of the Fortuin-Kasteleyn-Swendsen-Wang representation and Monte Carlo algorithm. *Physical Review D*, 38(6):2009–2012, 1988.
12. G. Ellison. Learning, local interaction, and coordination. *Econometrica: Journal of the Econometric Society*, pages 1047–1071, 1993.
13. J. Felsenstein. *Inferring Phylogenies*, volume 2. Sinauer Associates Sunderland, MA, 2004.
14. J. A. Fill and J. Kahn. Comparison inequalities and fastest-mixing Markov chains. *The Annals of Applied Probability*, 23(5):1778–1816, 2013.
15. A. Galanis, D. Štefankovič, and E. Vigoda. Swendsen-Wang algorithm on the mean-field Potts model. In *Proceedings of the 19th International Workshop on Randomization and Computation (RANDOM)*, pages 815–828, 2015.
16. D. Gamarnik and D. Katz. Correlation decay and deterministic FPTAS for counting list-colorings of a graph. In *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1245–1254, 2007.
17. S. Geman and C. Graffigne. Markov random field image models and their applications to computer vision. In *Proceedings of the International Congress of Mathematicians*, pages 1496–1517, 1986.
18. R. Gheissari, E. Lubetzky, and Y. Peres. Exponentially slow mixing in the mean-field Swendsen-Wang dynamics. In *Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1981–1988, 2018.
19. W. R. Gilks, S. Richardson, and D. J. Spiegelhalter. *Markov chain Monte Carlo in practice*. London: Chapman and Hall, 1996.

20 V. K. Gore and M. R. Jerrum. The Swendsen-Wang process does not always mix rapidly. *Journal of Statistical Physics*, 97(1-2):67–86, 1999.

21 G. R. Grimmett. *The Random-Cluster Model*, volume 333. Springer-Verlag, 2009.

22 H. Guo and M. Jerrum. Random cluster dynamics for the Ising model is rapidly mixing. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1818–1827, 2017.

23 T. P. Hayes and A. Sinclair. A general lower bound for mixing of single-site dynamics on graphs. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 511–520, 2005.

24 A. E. Holroyd. Some Circumstances Where Extra Updates Can Delay Mixing. *Journal of Statistical Physics*, 145(6):1649–1652, 2011.

25 E. Ising. Beitrag zur theorie des ferromagnetismus. *Zeitschrift für Physik*, 31(1):253–258, 1925.

26 M. Jerrum and A. Sinclair. Polynomial-time approximation algorithms for the Ising model. *SIAM Journal on Computing*, 22(5):1087–1116, 1993.

27 M. Jerrum, A. Sinclair, and E. Vigoda. A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries. *Journal of the ACM*, 51(4):671–697, 2004.

28 M. R. Jerrum, L. G. Valiant, and V. V. Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theoretical Computer Science*, 43(2-3):169–188, 1986.

29 R. Kannan, L. Lovász, and M. Simonovits. Random walks and an $O^*(n^5)$ volume algorithm for convex bodies. *Random Structures & Algorithms*, 11(1):1–50, 1997.

30 D. P. Landau and K. A. Binder. *A guide to Monte Carlo simulations in statistical physics*. Cambridge University Press, 2014.

31 W. Lenz. Beiträge zum verstandnis der magnetischen eigenschaften in festen korpern. *Physikalische Zeitschrift*, 21:613–615, 1920.

32 D. A. Levin and Y. Peres. *Markov Chains and Mixing Times, 2nd edition*, volume 107. American Mathematical Society, 2017.

33 L. Li, P. Lu, and Y. Yin. Correlation decay up to uniqueness in spin systems. In *Proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 67–84, 2013.

34 Y. Long, A. Nachmias, W. Ning, and Y. Peres. A power law of order 1/4 for critical mean-field Swendsen-Wang dynamics. *Memoirs of the American Mathematical Society*, 232(1092):84 pages, 2014.

35 F. Martinelli and E. Olivieri. Approach to equilibrium of Glauber dynamics in the one phase region. I. The attractive case. *Communications in Mathematical Physics*, 161(3):447–486, 1994.

36 F. Martinelli and E. Olivieri. Approach to equilibrium of Glauber dynamics in the one phase region. II. The general case. *Communications in Mathematical Physics*, 161(3):458–514, 1994.

37 A. Montanari and A. Saberi. The spread of innovations in social networks. *Proceedings of the National Academy of Sciences*, 107(47):20196–20201, 2010.

38 E. Mossel and A. Sly. Exact thresholds for Ising–Gibbs samplers on general graphs. *The Annals of Probability*, 41(1):294–328, 2013.

39 Y. Peres. Personal communication, 2016.

40 Y. Peres and P. Winkler. Can Extra Updates Delay Mixing? *Communications in Mathematical Physics*, 323(3):1007–1016, 2013.

41 C. J. Preston. Gibbs States on Countable Sets. *Cambridge Tracts in Mathematics*, 1974.

42 J. Salas and A. D. Sokal. Dynamic critical behavior of a Swendsen-Wang-type algorithm for the Ashkin-Teller model. *Journal of Statistical Physics*, 85(3-4):297–361, 1996.

43 J. Salas and A. D. Sokal. Dynamic critical behavior of the Swendsen-Wang algorithm: The two-dimensional three-state potts model revisited. *Journal of Statistical Physics*, 87(1-2):1–36, 1997.

44 A. Sinclair, P. Srivastava, D. Štefankovič, and Y. Yin. Spatial mixing and the connective constant: Optimal bounds. *Probability Theory and Related Fields*, 168(1-2):153–197, 2017.

45 A. Sinclair, P. Srivastava, and M. Thurley. Approximation algorithms for two-state anti-ferromagnetic spin systems on bounded degree graphs. *Journal of Statistical Physics*, 155(4):666–686, 2014.

46 A. Sly. Computational transition at the uniqueness threshold. In *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 287–296, 2010.

47 A. Sly and N. Sun. The computational hardness of counting in two-spin models on d-regular graphs. In *Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 361–369, 2012.

48 D. Štefankovič, S. Vempala, and E. Vigoda. Adaptive Simulated Annealing: A Near-optimal Connection between Sampling and Counting. *Journal of the ACM*, 56(3):1–36, 2009.

49 R. H. Swendsen and J. S. Wang. Nonuniversal critical dynamics in Monte Carlo simulations. *Physical Review Letters*, 58(2):86–88, 1987.

50 M. Ullrich. Rapid mixing of Swendsen-Wang and single-bond dynamics in two dimensions. *Dissertationes Mathematicae*, 2014.

51 J. S. Wang. Critical dynamics of the Swendsen-Wang algorithm in the three-dimensional Ising model. *Physica A*, 164(2):240–244, 1990.

52 D. Weitz. Counting independent sets up to the tree threshold. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC)*, pages 140–149, 2006.

A Proof of Theorem 11

In [38], Mossel and Sly show that ASSM (see Definition 9) implies optimal $O(n \log n)$ mixing of the Glauber dynamics on any n-vertex graph of bounded degree [23]. Our proof of Theorem 11 follows the approach in [38]. The key new novelty is the use of Theorem 7.

Proof of Theorem 11. Let $\{X_t^+\}$, $\{X_t^-\}$ be two instances of the chain such that X_0^+ is the "all plus" configuration and X_0^- is the "all minus" one. Since the chain is monotone there exists a monotone grand coupling of $\{X_t^+\}$ and $\{X_t^-\}$ such that $X_t^+ \geq X_t^-$ for all $t \geq 0$. The existence of a monotone grand coupling implies that the extremal "all plus" and "all minus" are the worst possible starting configurations, and thus

$$T_{\text{mix}}(P, \varepsilon) \leq T_{\text{coup}}(\varepsilon)$$

where $T_{\text{coup}}(\varepsilon)$ is the minimum t such that $\Pr[X_t^+ \neq X_t^-] \leq \varepsilon$, assuming $\{X_t^+\}$ and $\{X_t^-\}$ are coupled using the monotone coupling. Hence, it is sufficient to find t such that for all $v \in V$

$$\Pr[X_t^+(v) \neq X_t^-(v)] \leq \frac{\varepsilon}{n},$$

since the result would follow from a union bound over the vertices.

Choose $R \in \mathbb{N}$ such that ASSM holds; see Lemma 10. Let $s \in \mathbb{N}$ be arbitrary and fixed. For each $v \in V$, we define two instances $\{Y_t^+\}$ and $\{Y_t^-\}$ of the censored chain that until time s evolves as the chain P and after time s it evolves according to $P_{B(v,R)}$. By assumption

$P_{B(v,R)}$ is also monotone, so the evolutions of $\{Y_t^+\}$ and $\{Y_t^-\}$ can be coupled as follows: up to time s, $\{Y_t^+\}$ and $\{Y_t^-\}$ are coupled by setting $Y_t^+ = X_t^+$ and $Y_t^- = X_t^-$ for all $0 \le t \le s$; for $t > s$ the monotone coupling for $P_{B(v,R)}$ is used. Then, we have $X_t^+ \ge X_t^-$ and $Y_t^+ \ge Y_t^-$ for all $t \ge 0$.

Since $P \le P_{B(v,R)}$ by assumption, and the distribution ν^+ (resp., ν^-) of X_0^+ (resp., X_0^-) is such that ν^+/μ (resp., ν^-/μ) is trivially increasing (resp., decreasing), Theorem 7 implies $Y_t^+ \succeq X_t^+$ and $X_t^- \succeq Y_t^-$ for all $t \ge 0$. Hence,

$$Y_t^+ \succeq X_t^+ \succeq X_t^- \succeq Y_t^-.$$

Thus,

$$\begin{aligned}\Pr[X_t^+(v) \neq X_t^-(v)] &= \Pr[X_t^+(v) = +] - \Pr[X_t^-(v) = +] \\ &\le \Pr[Y_t^+(v) = +] - \Pr[Y_t^-(v) = +] \\ &= \Pr[Y_t^+(v) \neq Y_t^-(v)],\end{aligned}$$

where the first and third equations follow from the monotonicity of $\{X_t^+\}$, $\{X_t^-\}$, $\{Y_t^+\}$ and $\{Y_t^-\}$ and the inequality from the fact that $Y_t^+ \succeq X_t^+$ and $Y_t^- \preceq X_t^-$.

Recall our earlier definitions of $B(v,R)$ as the ball of radius R and $S(v,R)$ as the external boundary of $B(v,R)$; i.e., $B(v,R) = \{u \in V : \mathrm{dist}(u,v) \le R\}$ and let $S(v,R) = B(v,R+1) \backslash B(v,R)$. For ease of notation let $A = B(v,R+1) = B(v,R) \cup S(v,R)$ and for $\sigma^+, \sigma^- \in \Omega_A$ let $\mathcal{F}_s(\sigma^+,\sigma^-)$ be the event $\{X_s^+(A) = \sigma^+, X_s^-(A) = \sigma^-\}$. Then, for $t > s$ we have

$$\begin{aligned}\Pr[Y_t^+(v) \neq Y_t^-(v) \mid \mathcal{F}_s(\sigma^+,\sigma^-)] \le &\Big|\Pr[Y_t^+(v) = + \mid \mathcal{F}_s(\sigma^+,\sigma^-)] - \mu(v = + \mid \tau^+)\Big| \\ &+ \Big|\mu(v = + \mid \tau^+) - \mu(v = + \mid \tau^-)\Big| \\ &+ \Big|\Pr[Y_t^-(v) = + \mid \mathcal{F}_s(\sigma^+,\sigma^-)] - \mu(v = + \mid \tau^-)\Big|,\end{aligned} \tag{11}$$

where $\mu = \mu_{G,\beta}$, $\tau^+ = \sigma^+(S(v,R))$ and $\tau^- = \sigma^-(S(v,R))$.

Observe that $\mu(\cdot \mid \tau^+)$ and $\mu(\cdot \mid \tau^-)$ are the stationary measures of $\{Y_t^+\}$ and $\{Y_t^-\}$ respectively, and recall that by assumption

$$\max_{\sigma \in \Omega} T_{\mathrm{mix}}(P_{B(v,R)}, \sigma) \le T.$$

Hence, for $t = s + T \log_4 \lceil 8|A| \rceil$, we have

$$\Big|\Pr[Y_t^+(v) = + \mid \mathcal{F}_s(\sigma^+,\sigma^-)] - \mu(v = + \mid \tau^+)\Big| \le \frac{1}{8|A|}, \tag{12}$$

and similarly

$$\Big|\Pr[Y_t^-(v) = + \mid \mathcal{F}_s(\sigma^+,\sigma^-)] - \mu(v = + \mid \tau^-)\Big| \le \frac{1}{8|A|}. \tag{13}$$

We bound next $|\mu(v = + \mid \tau^+) - \mu(v = + \mid \tau^-)|$. For $u \in S(v,R)$, let a_u be defined as in (5) and let $S(v,R) = \{u_1, u_2, \ldots, u_l\}$ with $l = |S(v,R)|$. Let $\tau_0, \tau_1, \ldots, \tau_l$ be a sequence of configurations on $S(v,R)$ such that $\tau_j(u_k) = \tau^+(u_k)$ for $j < k \le l$ and $\tau_j(u_k) = \tau^-(u_k)$ for

$1 \le k \le j$. That is, $\tau_0 = \tau^+$, $\tau_l = \tau^-$ and τ_j is obtained from τ_{j-1} by changing the spin of u_j from $\tau^+(u_j)$ to $\tau^-(u_j)$. The triangle inequality then implies that

$$\begin{aligned}\Big|\mu(v = + \mid \tau^+) - \mu(v = + \mid \tau^-)\Big| &\le \sum_{j=1}^{l} \Big|\mu(v = + \mid \tau_{j-1}) - \mu(v = + \mid \tau_j)\Big| \\ &\le \sum_{j=1}^{l} \mathbb{1}\{\tau^+(u_j) \neq \tau^-(u_j)\} \cdot a_{u_j} \\ &= \sum_{u \in S(v,R)} \mathbb{1}\{\sigma^+(u) \neq \sigma^-(u)\} \cdot a_u. \qquad (14)\end{aligned}$$

Hence, plugging (12), (13) and (14) into (11), we get

$$\Pr[Y_t^+(v) \neq Y_t^-(v) \mid \mathcal{F}_s(\sigma^+, \sigma^-)] \le \frac{1}{4|A|} + \sum_{u \in S(v,R)} \mathbb{1}\{\sigma^+(u) \neq \sigma^-(u)\} \cdot a_u.$$

Now, if $X_s^+(A) = X_s^-(A)$, then $Y_t^+(A) = Y_t^-(A)$ for all $t \ge s$. Therefore,

$$\begin{aligned}&\Pr[Y_t^+(v) \neq Y_t^-(v)] = \sum_{\sigma^+ \neq \sigma^- \in \Omega_A} \Pr[Y_t^+(v) \neq Y_t^-(v) \mid \mathcal{F}_s(\sigma^+, \sigma^-)] \Pr[\mathcal{F}_s(\sigma^+, \sigma^-)] \\ &\le \frac{\Pr[X_s^+(A) \neq X_s^-(A)]}{4|A|} + \sum_{\sigma^+ \neq \sigma^- \in \Omega_A} \sum_{u \in S(v,R)} \mathbb{1}\{\sigma^+(u) \neq \sigma^-(u)\} \cdot a_u \cdot \Pr[\mathcal{F}_s(\sigma^+, \sigma^-)] \\ &= \frac{\Pr[X_s^+(A) \neq X_s^-(A)]}{4|A|} + \sum_{u \in S(v,R)} \Pr[X_s^+(u) \neq X_s^-(u)] \cdot a_u.\end{aligned}$$

By union bound,

$$\frac{\Pr[X_s^+(A) \neq X_s^-(A)]}{4|A|} \le \frac{1}{4|A|} \sum_{u \in A} \Pr[X_s^+(u) \neq X_s^-(u)] \le \frac{1}{4} \max_{u \in V} \Pr[X_s^+(u) \neq X_s^-(u)].$$

Moreover, the ASSM property (see Lemma 10) implies that

$$\begin{aligned}\sum_{u \in S(v,R)} \Pr[X_s^+(u) \neq X_s^-(u)] \cdot a_u &\le \max_{u \in V} \Pr[X_s^+(u) \neq X_s^-(u)] \sum_{u \in S(v,R)} a_u \\ &\le \frac{1}{4} \max_{u \in V} \Pr[X_s^+(u) \neq X_s^-(u)].\end{aligned}$$

Thus, we conclude that for every $v \in V$

$$\Pr[X_t^+(v) \neq X_t^-(v)] \le \Pr[Y_t^+(v) \neq Y_t^-(v)] \le \frac{1}{2} \max_{u \in V} \Pr[X_s^+(u) \neq X_s^-(u)]$$

for $t = s + T \log_4 \lceil 8|A| \rceil$. Taking the maximum over v

$$\max_{v \in V} \Pr[X_t^+(v) \neq X_t^-(v)] \le \frac{1}{2} \max_{v \in V} \Pr[X_s^+(v) \neq X_s^-(v)].$$

Iteratively, we get that for $\hat{T} = T \log_4 \lceil 8|A| \rceil \log_2 \lceil \frac{n}{\varepsilon} \rceil$

$$\max_{v \in V} \Pr[X_{\hat{T}}^+(v) \neq X_{\hat{T}}^-(v)] \le \frac{\varepsilon}{n}.$$

This implies that $T_{\text{mix}}(P, \varepsilon) \le T \log_4 \lceil 8|A| \rceil \log_2 \lceil \frac{n}{\varepsilon} \rceil$, so taking $\varepsilon = 1/4$ it follows that $T_{\text{mix}}(P) = O(T \log n)$ as desired. Moreover, since for $\varepsilon > 0$

$$(T_{\text{rel}}(P) - 1) \log(2\varepsilon)^{-1} \le T_{\text{mix}}(P, \varepsilon),$$

taking $\varepsilon = n^{-1}$ yields that $T_{\text{rel}}(P) = O(T)$; see Theorem 12.5 in [32]. ◀

B Proof of Theorem 7

Proof of Theorem 7. By assumption, X_t has distribution νP^t while $\hat{X}_t$ has distribution $\nu\hat{P}^t$ where $\hat{P}^t = P_{A_1} \dots P_{A_t}$. Since $\{P_A\}_{A\subseteq V}$ is a censoring for P, we have $P \le P_A$ for all $A \subseteq V$. We show first that this implies $P^t \le \hat{P}^t$.

Recall that P_{A_i} may be viewed as an operator from $L_2(\mu)$ to $L_2(\mu)$. The reversibility of P_{A_i} w.r.t. μ implies that P_{A_i} is self-adjoint; i.e., $P^*_{A_i} = P_{A_i}$. Also, since P is monotone, $P^k f$ is increasing for any integer $k > 0$ and any increasing function f; see Proposition 22.7 in [32]. Combining these facts, we have that for any pair of increasing positive functions $f, g : \mathbb{R}^{|\Omega|} \to \mathbb{R}$

$$\langle f, P^t g\rangle_\mu = \langle f, P(P^{t-1}g)\rangle_\mu \le \langle f, P_{A_1}(P^{t-1}g)\rangle_\mu = \langle P_{A_1} f, P^{t-1} g\rangle_\mu.$$

Note also that P_{A_1} is monotone, so $P_{A_1} f$ is increasing. Iterating this argument, we obtain

$$\langle f, P^t g\rangle_\mu \le \langle P_{A_1} f, P^{t-1} g\rangle_\mu \le \cdots \le \langle P_{A_t} \dots P_{A_1} f, g\rangle_\mu = \langle f, \hat{P}^t g\rangle_\mu.$$

This shows that $P^t \le \hat{P}^t$.

To prove $X_t \preceq \hat{X}_t$, we need to show that for any increasing function g

$$\sum_{\sigma\in\Omega} \nu P^t(\sigma) g(\sigma) \le \sum_{\sigma\in\Omega} \nu\hat{P}^t(\sigma) g(\sigma). \tag{15}$$

Let $h : \mathbb{R}^{|\Omega|} \to \mathbb{R}$ be the function given by $h(\tau) = \nu(\tau)/\mu(\tau)$ for $\tau \in \Omega$. Then we have

$$\begin{aligned}\sum_{\sigma\in\Omega} \nu P^t(\sigma) g(\sigma) &= \sum_{\sigma\in\Omega} \Big(\sum_{\tau\in\Omega} \nu(\tau) P^t(\tau,\sigma)\Big) g(\sigma) = \sum_{\sigma,\tau\in\Omega} \nu(\tau) P^t(\tau,\sigma) g(\sigma) \\ &= \sum_{\sigma,\tau\in\Omega} \mu(\tau) P^t(\tau,\sigma) g(\sigma) h(\tau) = \langle h, P^t g\rangle_\mu.\end{aligned}$$

Similarly,

$$\sum_{\sigma\in\Omega} \nu\hat{P}^t(\sigma) g(\sigma) = \langle h, \hat{P}^t g\rangle_\mu.$$

The function h is increasing by assumption, and thus (15) follows immediately from the fact that $P^t \le \hat{P}^t$. This establishes part 1 of the theorem. Part 2 of the theorem follows from part 1 and Lemma 2.4 in [40]. ◀

Sampling in Uniqueness from the Potts and Random-Cluster Models on Random Regular Graphs

Antonio Blanca[1]
School of Computer Science, Georgia Institute of Technology, Atlanta GA 30332, USA
ablanca@cc.gatech.edu

Andreas Galanis[2]
Department of Computer Science, University of Oxford, Parks Road, Oxford, OX1 3QD, UK
andreas.galanis@cs.ox.ac.uk

Leslie Ann Goldberg[2]
Department of Computer Science, University of Oxford, Parks Road, Oxford, OX1 3QD, UK
leslie.goldberg@cs.ox.ac.uk

Daniel Štefankovič[3]
Department of Computer Science, University of Rochester, Rochester, NY 14627, USA
stefanko@cs.rochester.edu

Eric Vigoda[1]
School of Computer Science, Georgia Institute of Technology, Atlanta GA 30332, USA
vigoda@cc.gatech.edu

Kuan Yang[2]
Department of Computer Science, University of Oxford, Parks Road, Oxford, OX1 3QD, UK
kuan.yang@cs.ox.ac.uk

Abstract

We consider the problem of sampling from the Potts model on random regular graphs. It is conjectured that sampling is possible when the temperature of the model is in the so-called uniqueness regime of the regular tree, but positive algorithmic results have been for the most part elusive. In this paper, for all integers $q \geq 3$ and $\Delta \geq 3$, we develop algorithms that produce samples within error $o(1)$ from the q-state Potts model on random Δ-regular graphs, whenever the temperature is in uniqueness, for both the ferromagnetic and antiferromagnetic cases.

The algorithm for the antiferromagnetic Potts model is based on iteratively adding the edges of the graph and resampling a bichromatic class that contains the endpoints of the newly added edge. Key to the algorithm is how to perform the resampling step efficiently since bichromatic classes can potentially induce linear-sized components. To this end, we exploit the tree uniqueness to show that the average growth of bichromatic components is typically small, which allows us to use correlation decay algorithms for the resampling step. While the precise uniqueness threshold on the tree is not known for general values of q and Δ in the antiferromagnetic case, our algorithm works throughout uniqueness regardless of its value.

In the case of the ferromagnetic Potts model, we are able to simplify the algorithm significantly by utilising the random-cluster representation of the model. In particular, we demonstrate

[1] Research supported in part by NSF grants CCF-1617306 and CCF-1563838.
[2] The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) ERC grant agreement no. 334828. The paper reflects only the authors' views and not the views of the ERC or the European Commission. The European Union is not liable for any use that may be made of the information contained therein.
[3] Research supported in part by NSF grant CCF-1563757.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018).
Editors: Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer; Article No. 33; pp. 33:1–33:15

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

that a percolation-type algorithm succeeds in sampling from the random-cluster model with parameters p, q on random Δ-regular graphs for all values of $q \geq 1$ and $p < p_c(q, \Delta)$, where $p_c(q, \Delta)$ corresponds to a uniqueness threshold for the model on the Δ-regular tree. When restricted to integer values of q, this yields a simplified algorithm for the ferromagnetic Potts model on random Δ-regular graphs.

2012 ACM Subject Classification Theory of computation → Randomness, geometry and discrete structures, Theory of computation → Design and analysis of algorithms

Keywords and phrases sampling, Potts model, random regular graphs, phase transitions

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.33

Related Version The full version is available at `https://arxiv.org/abs/1804.08111`. The theorem numbering here matches the full version.

1 Introduction

Random constraint satisfaction problems have been thoroughly studied in computer science in an effort to analyse the limits of satisfiability algorithms and understand the structure of hard instances. Analogously, understanding spin systems on random graphs [22, 23, 28, 3, 21, 7, 8] gives insights about the complexity of counting and the efficiency of approximate sampling algorithms. In this paper, we design approximate sampling algorithms for the Potts model on random regular graphs.

The Potts model is a fundamental spin system studied in statistical physics and computer science. The model has two parameters: an integer $q \geq 3$, which represents the number of states/colours of the model, and a real parameter $B > 0$, which corresponds to the so-called "temperature". We denote the set of colours by $[q] := \{1, \ldots, q\}$. For a graph $G = (V, E)$, configurations of the model are all possible assignments of colours to the vertices of the graph. Each assignment $\sigma : V \to [q]$ has a weight $w_G(\sigma)$ which is determined by the number $m(\sigma)$ of monochromatic edges under σ; namely, $w_G(\sigma) = B^{m(\sigma)}$. The Gibbs distribution μ_G is defined on the space of all configurations σ by

$$\mu_G(\sigma) = B^{m(\sigma)}/Z_G, \text{ where } Z_G = \textstyle\sum_\sigma B^{m(\sigma)}.$$

We also refer to μ_G as the Potts distribution; the quantity Z_G is known as the partition function. Well-known models closely related to the Potts model are the Ising and colourings models. The Ising model is the special case $q = 2$ of the Potts model, while the q-colourings model is the "zero-temperature" case $B = 0$ of the Potts model, where the distribution is supported on the set of proper q-colourings.

The behaviour of the Potts model has significant differences depending on whether B is less or larger than 1. When $B < 1$, configurations where most neighbouring vertices have different colours have large weight and the model is called *antiferromagnetic*; in contrast, when $B > 1$, configurations where most neighbouring vertices have the same colours have large weight and the model is called *ferromagnetic*. One difference between the two cases that will be relevant later is that the ferromagnetic Potts model admits a random-cluster representation – the details of this representation are given in Section 2.1.

Sampling from the Potts model is a problem that is frequently encountered in running simulations in statistical physics or inference tasks in computer science. To determine the efficiency and accuracy of sampling methods, it is relevant to consider the underlying phase

transitions, which signify abrupt changes in the properties of the Gibbs distribution when the underlying parameter changes. The so-called uniqueness phase transition captures the sensitivity of the state of a vertex to fixing far-away boundary conditions. As an example, in the case of the ferromagnetic Potts model on the Δ-regular tree, uniqueness holds when root-to-leaves correlations in the Potts distribution vanish as the height of the tree goes to infinity; it is known that this holds iff $B < B_c(q, \Delta)$, where $B_c(q, \Delta)$ is the "uniqueness threshold" (cf. (2) for its value). Connecting the uniqueness phase transition with the performance of algorithms is a difficult task that is largely under development. This connection is well-understood on the grid, where it is known that the mixing time of local Markov chains, such as the Glauber dynamics, switches from polynomial to exponential at the corresponding uniqueness threshold, see for example [18, 17, 27, 1, 19, 2].

For random Δ-regular graphs or, more generally, graphs with maximum degree Δ, the uniqueness threshold on the Δ-regular tree becomes relevant. For certain two-state models, such as the ferromagnetic Ising model and the hard-core model, it has been proved that Glauber dynamics mixes rapidly when the underlying parameter is in uniqueness and that the dynamics mixes slowly otherwise, see [22, 23, 8]. The same picture is conjectured to hold for the Potts model as well, but this remains open. More generally, there has been significant progress in understanding the complexity of sampling from the Gibbs distribution in two-state systems, but for multi-state systems progress has been slower, especially on the algorithmic side.

In this paper, for all integers $q \geq 3$ and $\Delta \geq 3$, we design approximate sampling algorithms for the q-state Potts model on random Δ-regular graphs (regular graphs with n vertices chosen uniformly at random), when the parameter B lies in the uniqueness regime of the regular tree, for both the ferromagnetic and antiferromagnetic cases. Our algorithms are not based on a Markov chain approach but proceed by iteratively adding the edges of the graph and performing a resampling step at each stage. As such, our algorithms can produce samples that are within error $1/n^{\delta}$ from the Potts distribution for some fixed constant $\delta > 0$ (which depends on B, q, Δ).

▶ Remark. There are certain "bad" Δ-regular graphs where the algorithms will fail to produce samples with the desired accuracy; saying that the algorithms work on random Δ-regular graphs means that the number of these "bad" graphs with n vertices is a vanishing fraction of all Δ-regular graphs with n vertices for large n. Moreover, we can recognise the "good" graphs (where our algorithms will successfully produce samples with the desired accuracy) in polynomial time.

Our approach is inspired by Efthymiou's algorithm [6, 7] for sampling q-colourings on $G(n, d/n)$; the algorithm there also proceeds by iteratively adding the edges of the graph and exploits the uniqueness on the tree to show that the sampling error is small. However, for the antiferromagnetic Potts model, the resampling step turns out to be significantly more involved and we need substantial amount of work to ensure that it can be carried out efficiently, as we explain in detail in Section 3. Nevertheless, for the ferromagnetic case, we manage to give a far simpler algorithm by utilising the random-cluster representation of the model (see Section 2.1). In particular, we demonstrate that a percolation-type algorithm succeeds in sampling approximately from the random-cluster model with parameters p, q on random Δ-regular graphs for all values of $q \geq 1$ and $p < p_c(q, \Delta)$, where $p_c(q, \Delta)$ corresponds to a uniqueness threshold for the model on the Δ-regular tree. When restricted to integer values of q, this yields a simple algorithm for the ferromagnetic Potts model on random Δ-regular graphs.

To conclude this introductory section, we remark that, for many antiferromagnetic spin systems on random graphs, typical configurations in the Gibbs distribution display absence of long-range correlations even beyond the uniqueness threshold, up to the so-called reconstruction threshold [20, 12]. Note that uniqueness guarantees the absence of long-range correlations under a "worst-case" boundary, while non-reconstruction only asserts the absence of long-range correlations under "typical" boundaries; it is widely open whether this weaker notion is in fact sufficient for sampling on random graphs. On an analogous note, for the ferromagnetic Potts model on random regular graphs, the structure of typical configurations can be fairly well understood using probabilistic arguments for all temperatures (see, e.g., [5, 11]) and it would be very interesting to exploit this structure for the design of sampling algorithms beyond the uniqueness threshold.

2 Definitions and Main Results

To formally state our results, we first review in Section 2.1 the definition of the random-cluster model. In Section 2.2, we state results from the literature about uniqueness on the regular tree for the Potts and random-cluster models. Then, in Section 2.3, we state our algorithmic results for the ferromagnetic Potts and random-cluster models and, in Section 2.4, our result for the antiferromagnetic Potts model.

2.1 The random-cluster model

The random-cluster model has two parameters $p \in [0,1]$ and $q > 0$; note that q in this case can take non-integer values. For a graph $G = (V, E)$, we denote the random-cluster distribution on G by φ_G; this distribution is supported on the set of all edge subsets. In particular, for $S \subseteq E$, let $k(S)$ be the number of connected components in the graph $G' = (V, S)$ (isolated vertices do count). Then,

$$\varphi_G(S) = \frac{p^{|S|}(1-p)^{|E\setminus S|}q^{k(S)}}{Z_G^{\mathrm{rc}}}, \text{ where } Z_G^{\mathrm{rc}} = \sum_{S\subseteq E} p^{|S|}(1-p)^{|E\setminus S|}q^{k(S)}.$$

Following standard terminology, each edge in S will be called open, while each edge in $E\setminus S$ closed. For integer values of q, the random-cluster and ferromagnetic Potts models are connected as follows.

▶ **Lemma 1** (see, e.g., [13]). *Let $q \geq 2$ be an integer, $B > 1$, and $p = 1 - 1/B$. Then, the following hold for any graph $G = (V, E)$.*

- *Let $S \subseteq E$ be distributed according to the RC distribution φ_G with parameters p, q. Consider the configuration σ obtained from S by assigning each component in the graph (V, S) a random colour from $[q]$ independently. Then, σ is distributed according to the Potts distribution μ_G with parameter B.*
- *Conversely, suppose that $\sigma : V \to [q]$ is distributed according to the Potts distribution μ_G with parameter B. Consider $S \subseteq E$ obtained by adding to S each monochromatic edge under σ with probability p independently. Then, S is distributed according to the RC distribution φ_G with parameters p, q.*

2.2 Uniqueness for Potts and random-cluster models on the tree

In this section, we review uniqueness on the tree for the Potts and random-cluster models.

We start with the Potts model. For a configuration σ and a set U, we denote by σ_U the restriction of σ to the set U; in the case of a single vertex u, we simply write σ_u to denote the colour of u. Denote by $\mathbb{T}_\Delta$ the infinite $(\Delta-1)$-ary tree with root vertex ρ and, for an integer $h \geq 0$, denote by T_h the subtree of $\mathbb{T}_\Delta$ induced by the vertices at distance $\leq h$ from ρ. Let L_h be the set of leaves of T_h.

▶ **Definition 2.** Let $B > 0$ and $q, \Delta \geq 3$ be integers. The q-state Potts model with parameter $B > 0$ has *uniqueness* on the infinite $(\Delta-1)$-ary tree if, for all colours $c \in [q]$, it holds that

$$\limsup_{h\to\infty} \max_{\tau: L_h \to [q]} \left| \mu_{T_h}\left(\sigma_\rho = c \mid \sigma_{L_h} = \tau\right) - \frac{1}{q} \right| = 0. \tag{1}$$

For the ferromagnetic q-state Potts model ($B > 1$), it is known that uniqueness holds on the $(\Delta-1)$-ary tree iff $B < B_c(q,\Delta)$, where

$$B_c(q,\Delta) = 1 + \inf_{y>1} h(y), \text{ where } h(y) := \frac{(y-1)(y^{\Delta-1}+q-1)}{y^{\Delta-1}-y}. \tag{2}$$

For the antiferromagnetic Potts model ($B < 1$), the uniqueness threshold on the tree is not yet known in full generality. It is known that the model does not have uniqueness when $B < \frac{\Delta-q}{\Delta}$ [10] and therefore $B \geq (\Delta-q)/\Delta$ is a necessary condition for uniqueness to hold. It is also conjectured that this condition is sufficient but this has only been established for small values of q and Δ [9]. In the case $q = 3$, [9] also established the uniqueness threshold for all Δ: for $\Delta \geq 4$, uniqueness holds iff $B \in [(\Delta-3)/\Delta, 1)$ and, for $\Delta = 3$ uniqueness holds iff $B \in (0,1)$. For the q-colourings model ($B = 0$), Jonasson [16], building on work of Brightwell and Winkler [4], established that the model has uniqueness iff $q > \Delta$.

Uniqueness for the random-cluster model on the tree is less straightforward to define. Häggström [14] studied uniqueness of random-cluster measures on the infinite $(\Delta-1)$-ary tree where all infinite components are connected "at infinity" – we review his results in more detail in Section 7.1 of the full version. He showed that, for all $q \geq 1$, a sufficient condition for uniqueness is that $p < p_c(q,\Delta)$, where $p_c(q,\Delta)$ is given by[4]:

$$p_c(q,\Delta) = 1 - \frac{1}{1+\inf_{y>1} h(y)}, \text{ where } h(y) := \frac{(y-1)(y^{\Delta-1}+q-1)}{y^{\Delta-1}-y}. \tag{3}$$

Note that the critical values in (2) and (3) are connected for integer values of q via $p_c(q,\Delta) = 1 - \frac{1}{B_c(q,\Delta)}$. Häggström [14] also conjectured that uniqueness for the random-cluster model holds on $\mathbb{T}_\Delta$ when $p > \frac{q}{q+\Delta-2}$ for all $q \geq 1$; this remains open but progress has been made in [15].

2.3 Sampling ferro Potts and random-cluster models on random regular graphs

We begin by stating our result for the random-cluster model on random regular graphs.

▶ **Theorem 6.** *Let $\Delta \geq 3$, $q \geq 1$ and $p < p_c(q,\Delta)$. Then, there exists a constant $\delta > 0$ such that, for all sufficiently large n, the following holds with probability $1 - o(1)$ over the choice of a random Δ-regular graph $G = (V,E)$ with n vertices.*

There is a polynomial-time algorithm which, on input the graph G, outputs a random set $S \subseteq E$ whose distribution ν_S is within total variation distance $O(1/n^\delta)$ from the RC distribution φ_G with parameters p, q, i.e., $\|\nu_S - \varphi_G\|_{\mathrm{TV}} = O(1/n^\delta)$.

[4] In [14], $p_c(q,\Delta)$ is defined in a different way, but the two definitions are equivalent for all $q \geq 1$.

For integer values of q, Theorem 6 combined with the translation between the random-cluster and Potts models (cf. Lemma 1) yields a sampling algorithm for the ferromagnetic q-state Potts model on random regular graphs. Since uniqueness for the ferromagnetic Potts model holds iff $B < B_c(q, \Delta)$ and $p_c(q, \Delta) = 1 - \frac{1}{B_c(q,\Delta)}$, we therefore have the following corollary of Theorem 6.

▶ **Corollary 7.** *Let $\Delta \geq 3$, $q \geq 3$ and $B > 1$ be in the uniqueness regime of the $(\Delta - 1)$-ary tree. Then, there exists a constant $\delta > 0$ such that, for all sufficiently large n, the following holds with probability $1 - o(1)$ over the choice of a random Δ-regular graph $G = (V, E)$ with n vertices.*

There is a polynomial-time algorithm which, on input the graph G, outputs a random assignment $\sigma : V \to [q]$ whose distribution ν_σ is within total variation distance $O(1/n^\delta)$ from the Potts distribution μ_G with parameter B, i.e., $\|\nu_\sigma - \mu_G\|_{\mathrm{TV}} = O(1/n^\delta)$.

2.4 Sampling antiferro Potts on random Δ-regular graphs

The algorithm of Corollary 7 for the ferromagnetic Potts model does not extend to the antiferromagnetic case since there is no analogous connection with the random-cluster model in this case. Nevertheless, we are able to design a sampling algorithm on random regular graphs when the parameter B is in uniqueness via a far more elaborate approach which consists of recolouring (large) bichromatic colour classes.

▶ **Theorem 8.** *Let $\Delta \geq 3$, $q \geq 3$ and $B \in (0, 1)$ be in the uniqueness regime of the $(\Delta-1)$-ary tree with $B \neq (\Delta - q)/\Delta$. Then, there exists a constant $\delta > 0$ such that, for all sufficiently large n, the following holds with probability $1 - o(1)$ over the choice of a random Δ-regular graph $G = (V, E)$ with n vertices.*

There is a polynomial-time algorithm which, on input the graph G, outputs a random assignment $\sigma : V \to [q]$ whose distribution ν_σ is within total variation distance $O(1/n^\delta)$ from the Potts distribution μ_G with parameter B, i.e., $\|\nu_\sigma - \mu_G\|_{\mathrm{TV}} = O(1/n^\delta)$.

Note, the algorithm in the antiferromagnetic case works throughout uniqueness apart from the single point $(\Delta - q)/\Delta$, where uniqueness on the tree is expected to hold but the model is conjectured to be at criticality.

3 Proof Approach

In this section, we outline the main idea behind the algorithms of Theorems 6 and 8, and the key obstacles that we have to address. We focus on the antiferromagnetic Potts model where the details are much more complex and discuss how we get the simplification for the ferromagnetic case via the random-cluster model later.

▶ **Definition 9.** For an n-vertex graph $G = (V, E)$ with maximum degree Δ, a cycle is *short* if its length is at most $\frac{1}{5} \log_{\Delta-1} n$, and is *long* otherwise.

Let G be a uniformly random Δ-regular graph with n vertices. Following the approach of Efthymiou [7], our algorithm starts from the subgraph of G consisting of all short cycles, which we denote by G'. It is fairly standard to show that, with probability $1 - o(1)$ over the choice of G, the subgraph G' is a *disjoint* union of short cycles, see Lemma 12. It is therefore possible to sample a configuration σ' on G' which is distributed according to the Potts distribution $\mu_{G'}$ (exactly). This can be accomplished in several ways; in fact, since the cycles are disjoint and each cycle has logarithmic length, this initial sampling step can even be done via brute force in polynomial time (though it is not hard to come up with much faster algorithms).

After this initial preprocessing, the algorithm then proceeds by adding sequentially the edges that do not belong to short cycles. At each step, the current configuration is updated with the aim to preserve its distribution close to the Potts distribution of the new graph (with the edge that we just added). Key to this update procedure is a resampling step which is performed only when the endpoints of a newly added edge $\{u, v\}$ happen to have the same colours under the current configuration; intuitively, some action is required in this case because the weight of the current configuration reduces by a factor of $B < 1$ in the new graph (because of the added edge). The resampling step consists of recolouring a bichromatic class.

▶ **Definition 10.** Let $G = (V, E)$ be a graph and $\sigma : V \to [q]$ be a configuration. For colours $c_1, c_2 \in [q]$, let $\sigma^{-1}(c_1, c_2)$ be the set of vertices that have either colour c_1 or colour c_2 under σ. For distinct colours $c_1, c_2 \in [q]$, we say that $U = \sigma^{-1}(c_1, c_2)$ is the (c_1, c_2)-colour-class of σ and that U is a bichromatic class under σ. We refer to a connected component of $G[U]$ as a bichromatic component.

In the proper colourings case ($B = 0$), Efthymiou [7] demonstrated that the resampling step when adding an edge $e = \{u, v\}$ can be done by just flipping the colours of a bichromatic component chosen uniformly at random among those containing one of the vertices u and v (say u). The rough idea there is that, when the colourings model is in uniqueness, the bichromatic components on a random graph are typically small in size. At the same time, by the initial preprocessing step, the edge $e = \{u, v\}$ does not belong to a short cycle and therefore u and v are far away in the graph without e. Hence, u and v are unlikely to belong to the same bichromatic component and the flipping step will succeed in giving u and v different colours with good probability.

Unfortunately, this flipping method does not work for the antiferromagnetic Potts model. It turns out that when $q < \Delta$ and even when the Potts model is in uniqueness, bichromatic components can be large and therefore u and v may belong to the same bichromatic component. To make matters worse, these bichromatic components can be quite complicated (with many short/long cycles) and we need a more elaborate approach in our setting to succeed in giving u and v different colours without introducing significant bias to the sampler.

The key to overcoming these obstacles lies in the observation that the assignment of the two colours in a bichromatic component follows the Ising distribution, see Observation 22 for the precise formulation. Hence we can hope to use an approximate sampling algorithm for the Ising model in the resampling step. The natural implementation of this idea however fails: known algorithms for the antiferromagnetic Ising model, based on correlation decay, work as long as $B > (\Delta - 2)/\Delta$, where Δ is the maximum degree of the graph [25, 29]. In general, this inequality is not satisfied for us, i.e., there exist B in the uniqueness regime such that $B < \frac{\Delta-2}{\Delta}$.

Fortunately, we can employ fairly recent technology for two-state models [22, 26, 24] which demonstrates that the graph parameter that matters is not actually the maximum degree of the graph but rather the "average growth" of the graph. While we cannot apply any of the existing results in the literature directly, adapting these ideas to the antiferromagnetic Ising model is fairly straightforward, using results from Mossel and Sly [22]. The more difficult part in our setting is proving that the average growth of the bichromatic components that we consider for resampling is indeed small for "typical" configurations σ (note that in the worst case, the whole graph can be a bichromatic class which has large average growth for our purposes, so a probability estimate over σ is indeed due). Let us first formalise the notion of average growth that we use.

▶ **Definition 11.** Let M, b be positive constants and $G = (V, E)$ be a graph with n vertices. We say that G has average growth b up to depth $L = \lceil M \log n \rceil$ if for all vertices $v \in V$ the total number of paths with L vertices starting from v is less than b^L.

The notion of average growth is similar to the notion of connective constant for finite graphs used in [26, 24], the reason for the slightly different definition is that we will need an explicit handle on the constant M controlling the depth. Note that since we only consider paths with a fixed logarithmic length, this places a lower bound on the accuracy of the sampling algorithm. Nevertheless, by choosing the constant M sufficiently large, this will still be sufficient to make the error of our sampler polynomially small. In particular, as long as the inequality $b\frac{1-B}{1+B} < 1$ is satisfied, for all sufficiently large M, we obtain an approximate sampler for the antiferromagnetic Ising model with parameter B on graphs of average growth b up to depth $L = \lceil M \log n \rceil$, see Theorem 24 for details.

The next stage is to bound the average growth of bichromatic classes. Here, we utilise the tree uniqueness and the tree-like structure of random Δ-regular graphs (cf. Lemma 15) to provide an upper bound on the probability that a path is bichromatic. For paths of logarithmic length L, we show in Lemma 31 that this probability is bounded above by K^L, where K is roughly $(1 + B)/(B + q - 1)$. Since in a Δ-regular graph there are at most $\Delta(\Delta - 1)^{L-2}$ paths with L vertices, we therefore obtain that, in most configurations σ, the average growth b of bichromatic components is bounded above by $(\Delta - 1)K$. When B is in uniqueness, we have that $B > (\Delta - q)/\Delta$, and therefore the inequality $b\frac{1-B}{1+B} < 1$ that is required for the Ising sampler to work is satisfied (quite tightly in fact).

The final piece is to bound the error that is introduced by the resampling steps; note that, even if the resampling steps were perfect, some error is always introduced as a result of the placement of the new edge which reweights the probability that u and v have different colours. The idea now is to use the correlation decay properties on bichromatic components to show that, in the graph without the edge $\{u, v\}$, the correlation between the colours of u and v is relatively small since in that graph they are far apart (recall that $\{u, v\}$ does not belong to a short cycle in G). In Lemma 25, we show that the correlation between u and v can be upper bounded as a weighted sum over paths connecting u and v. Unfortunately, it is not possible to ensure that the correlation is small for each step separately since the natural union bound does not work. The right way to control the error of the sampler is to aggregate over all steps; we thus obtain a bound on the error via a weighted sum over logarithmically long cycles of the random graph G. Using a simple expectation argument of this latter quantity (and Markov's inequality), we obtain that the error will be small with probability $1 - o(1)$ over the choice of the graph G.

The algorithm that we described for the antiferromagnetic Potts model can actually be adapted to the ferromagnetic case as well. However, as mentioned earlier, we follow a different (and surprisingly simpler) route using the random-cluster representation of the model. At a very rough level, the reason behind the simplification is that the components in the random-cluster model provide a much better grip on capturing the properties of the Potts distribution than the bichromatic-component proxy we used earlier. Indeed, just as we described in the antiferromagnetic case, bichromatic components for the ferromagnetic Potts model can also be linear-sized. However, once we translate the Potts configuration to its random-cluster representation (cf. Lemma 1), the components in the latter are small in size (when the model is in the uniqueness region $p < p_c(q, \Delta)$) and therefore vertices that are far away do not belong to the same component. This allows us to perform the resampling step in the random-cluster model by a simple percolation procedure. The details can be found in Section 5.

4 Properties of random regular graphs

In this section, we state and prove structural properties of random Δ-regular graphs which ensure that our algorithms for the random-cluster and Potts models have the desired accuracy (cf. Remark 1). The proofs of these lemmas are fairly standard and are obtained by working in the well-known configuration model (see full version for details). The following lemma guarantees that short cycles are disjoint in a random Δ-regular graph.

▶ **Lemma 12.** *Let $\Delta \geq 3$ be an integer. Then, with probability $1 - o(1)$ over the choice of a uniformly random Δ-regular graph with n vertices, any two distinct cycles of length $\leq \frac{1}{5}\log_{\Delta-1} n$ are disjoint, i.e., they do not share any common vertices or edges.*

The next lemma guarantees that certain weighted sums over cycles are small; this bound will be used to show that the aggregate error of our samplers is small (cf. Section 3).

▶ **Lemma 13.** *Let $\Delta \geq 3$. Then, for any constant $W > \Delta - 1$ and any constant $\ell_0 > 0$, there exists a constant $\delta > 0$ such that the following holds with probability $1 - O(1/n^{\delta})$ over the choice of $G \sim \mathcal{G}_{n,\Delta}$. Let C_ℓ denote the number of cycles of length ℓ. Then, $\sum_{\ell \geq \ell_0 \log n} \ell C_\ell / W^\ell \leq 1/(2n^{\delta})$.*

Our next lemma captures the tree-like structure of random Δ-regular graphs that will be relevant for us. In particular, we give a description of the neighbourhood structure around a path. To do this accurately, we will need a few definitions. Let $G = (V, E)$ be a graph. For a vertex $v \in V$ and integer $h \geq 0$, we denote by $\Gamma_h(G, v)$ the set of vertices at distance $\leq h$ from v.

▶ **Definition 14.** Let G be a graph and P be a path in G with vertices $u_1, \ldots, u_\ell$. Let $G \backslash P$ be the graph obtained from G by removing the edges of the path P. Then, for an integer $h \geq 0$, the *h-graph-neighbourhood* of the path P is the subgraph of $G \backslash P$ induced by the vertex set $\bigcup_{i \in [\ell]} \Gamma_h(G \backslash P, u_i)$. A connected component of the h-graph-neighbourhood will be called *isolated* if it contains *exactly one* of the vertices $u_1, \ldots, u_\ell$.

▶ **Lemma 15.** *Let $\Delta \geq 3$. Then, for any constant integer $h \geq 0$ and any $\epsilon > 0$, there exists a constant $\ell_1 > 0$ such that the following holds for all sufficiently large n.*

With probability $1 - O(1/n^2)$ over the choice of $G \sim \mathcal{G}_{n,\Delta}$, every path P in G with ℓ vertices with $\ell_1 \leq \ell \leq n^{9/10}$ has an h-graph-neighbourhood with at least $(1 - \epsilon)\ell$ isolated tree components.

To conclude this section, we clarify a small point relevant to Remark 1. We will only utilise Lemma 15 for paths of logarithmic length (despite that the lemma is stated for convenience for much longer paths) and therefore the property can be checked in polynomial time. Similarly, the sum in Lemma 13 will only be considered for cycles of logarithmic length and hence the (restricted) inequality can also be checked in polynomial time.

5 Algorithm for the random-cluster model

Theorem 6 is obtained by analysing the following algorithm when the input is a random Δ-regular graph $G = (V, E)$. The detailed description of the algorithm is given in Figure 1 of the full version.

Theorem 21 of the full version details the performance of the algorithm when the input is a random Δ-regular graph; the statement is analogous to that of Theorem 6 so we omit it from this short version.

$E' := \{e \in E \mid e \text{ belongs to a short cycle}\}$
if $G' = (V, E')$ contains a component which is neither a cycle nor an isolated vertex **then** FAIL
else { Sample an RC configuration $S' \subseteq E'$ on G' (according to $\varphi_{G'}$);
Add to S' each edge in $E \backslash E'$ independently with probability $p/(q + (1-p)q)$;
Output the resulting set $S \subseteq E$ }

To motivate the algorithm, we first display how to update an RC configuration when we add a single edge $\{u, v\}$. In fact, to control the effect of adding an edge, we need a bound on the probability of the event that there is an open path between u and v; we denote this event by $u \leftrightarrow v$.

▶ **Lemma 17.** *Let $p \in (0,1)$ and $q \geq 1$, and consider arbitrary $\epsilon \in (0, 1/q)$.*

Let $G = (V, E)$ be a graph and u, v be two vertices such that $\{u, v\} \notin E$ and $\varphi_G(u \leftrightarrow v) \leq \epsilon$. Consider the graph $G' = (V, E')$ obtained from G by adding the edge $\{u, v\}$. Sample a random subset of edges $Y \subseteq E'$ as follows: first, sample a subset of edges $X \subseteq E$ according to the RC measure φ_G and, then, set $Y = X \cup \{e\}$ with probability $p/(q + (1-p)q)$, and $Y = X$ otherwise.

Then, the distribution of Y, denoted by ν_Y, is within total variation distance $2q\epsilon$ from the RC distribution $\varphi_{G'}$ on G' with parameters p, q, i.e., $\|\nu_Y - \varphi_{G'}\|_{\mathrm{TV}} \leq 2q\epsilon$.

To give the intuition behind the sampling procedure of Lemma 17, suppose that u and v were in distinct components of G. Then $\varphi_G(u \leftrightarrow v) = 0$ and the sampling procedure in Lemma 17 would actually produce an exact sample from $\varphi_{G'}$. In the setting of Lemma 17, we do not have this ideal situation but we have a close analogue, i.e., the probability of the event that u and v belong to the same component in a random-cluster configuration is less than ϵ.

To utilise Lemma 17, we need to upper bound the probability that two vertices belong to the same component in a RC configuration. In turn, it suffices to bound the probability that there is an open path between the vertices. To this end, we utilise the uniqueness and the tree-like structure around paths (cf. Definition 14) to show the following.

▶ **Lemma 18.** *Let $\Delta \geq 3$ be an integer, $q \geq 1$ and $p < p_c(q, \Delta)$. There exist constants $K < 1/(\Delta - 1)$ and $\epsilon > 0$ such that the following holds for all sufficiently large integers ℓ and h. Let G be a Δ-regular graph and P be a path with ℓ vertices whose h-graph-neighbourhood contains $(1-\epsilon)\ell$ isolated tree components. Let φ_G be the RC distribution on G with parameters p, q. Then, $\varphi_G(\text{path } P \text{ is open}) \leq K^\ell$.*

The proof of Lemma 18 is the most intricate part of our analysis for the random-cluster model algorithm. The difficulty is that the random-cluster model cannot be treated as a spin system due to the dependence of the model on the number of components. The rough intuition behind the proof is that, in the graph without the edges of the path P, the tree-like structure combined with uniqueness on the tree imply that (most) vertices in P belong to distinct components. As a result, when we add the edges of the path P, the trees decorating the path P do not have a significant effect on the number of components that vertices in P belong to and can be treated essentially as "not being there" in the analysis. Of course, the actual argument needs to account for all these effects quite carefully, see also Section 7.2 of the full version.

Using monotonicity properties of the random-cluster model (see Lemma 19 in the full version), we can extend the bound in Lemma 18 to arbitrary subgraphs of a target graph G. In particular, suppose that G, P are as in Lemma 18 and that G' is a subgraph of G which contains the path P. Then it also holds that $\varphi_{G'}(P \text{ is open}) \leq K^\ell$. Using this and Lemma 13, we obtain the following.

▶ **Lemma 20.** *Let $\Delta \geq 3$ be an integer, $q \geq 1$ and $p < p_c(q, \Delta)$. Then, there exists a constant $\delta > 0$ such that, for all sufficiently large n, the following holds with probability $1 - o(1)$ over the choice of a uniformly random Δ-regular graph $G = (V, E)$ with n vertices.*

Let $e_1, \ldots, e_t$ be the edges of G that do not belong to short cycles. For $j \in [t]$, let $e_j = \{u_j, v_j\}$ and G_j be the subgraph $G \backslash \{e_1, \ldots, e_j\}$. Then, it holds that $\sum_{j=1}^{t} \varphi_{G_j}(u_j \leftrightarrow v_j) \leq 1/n^{\delta}$.

Combining Lemmas 17 and 20, the proof of Theorem 6 is concluded in Section 5.3 of the full version.

6 Algorithm for the antiferromagnetic Potts model

In this section, following the outline of Section 3, we give some more precise technical details for the algorithm in the case of the antiferromagnetic Potts model (see also the relevant Section 6 of the full version).

6.1 Connection between Potts on bichromatic classes and the Ising model.

Our sampling algorithm for the Potts model uses an approximate sampler for the Ising model as a subroutine. Recall that the Ising model is the special case $q = 2$ of the Potts model; to distinguish between the models, we will use π_G to denote the Ising distribution on G with parameter B. Sometimes we will need to replace the binary set of states $\{1, 2\}$ in the Ising model by other binary sets to facilitate the arguments; we use $\pi_G^{c_1,c_2}$ to denote the Ising distribution with binary set of states $\{c_1, c_2\}$ (we will have that $c_1, c_2 \in [q]$). For vertices u, v, we also write $\pi_{G,u,v}^{c_1,c_2}$ to denote the Ising distribution on G conditioned on u taking the state c_1 and v the state c_2. The following observation details the connection between the distribution of the Potts model on bichromatic classes and the Ising model.

▶ **Observation 22.** *Let $q \geq 3$ and $B > 0$. Let $G = (V, E)$ be a graph, U be a subset of V and c, c' be distinct colours in $[q]$. Then, for any configuration $\eta : U \to \{c, c'\}$, it holds that $\mu_G\big(\sigma_U = \eta \mid \sigma^{-1}(c, c') = U\big) = \pi_{G[U]}^{c,c'}(\eta)$, i.e., conditioned on U being the (c, c')-colour-class in the Potts distribution μ_G, the marginal distribution on U is the Ising distribution $\pi_{G[U]}$ (with set of states $\{c, c'\}$).*

Adapting results of [22], we show the following algorithm for the Ising model in Section 9 of the full version.

▶ **Theorem 24.** *Let $B \in (0, 1)$ and $b > 0$ be constants such that $b\frac{1-B}{1+B} < 1$, and let $\Delta \geq 3$ be an integer. Then, there exists $M_0 > 0$ such that the following holds for all $M > M_0$.*

There is a polynomial-time algorithm that, on input an n-vertex graph G with maximum degree at most Δ and average growth b up to depth $L = \lceil M \log n \rceil$, outputs a configuration $\tau : V \to \{1, 2\}$ whose distribution ν_τ is within total variation distance $1/n^{10}$ from the Ising distribution on G with parameter B, i.e., $\|\nu_\tau - \pi_G\|_{\mathrm{TV}} \leq 1/n^{10}$.

Moreover, the algorithm, when given as additional input two vertices u and v in G, outputs a configuration $\tau : V \to \{1, 2\}$ such that $\tau_u = 1$ and $\tau_v = 2$, and whose distribution ν_τ satisfies $\|\nu_\tau - \pi_{G,u,v}(\cdot)\|_{\mathrm{TV}} \leq 1/n^{10}$, where $\pi_{G,u,v}$ is the Ising distribution on G conditioned on u having state 1 and v having state 2.

In addition, we will use the following spatial mixing result to analyse the accuracy of our algorithm for the antiferromagnetic Potts model. The proof is given in Section 9.3 of the full version.

▶ **Lemma 25.** *Let $B \in (0,1)$ and $b > 0$ be constants such that $b\frac{1-B}{1+B} < 1$. Then, there exists $M_0' > 0$ such that the following holds for all $M > M_0'$. Let G be an n-vertex graph with average growth b up to depth $L = \lceil M \log n \rceil$, and let u, v be distinct vertices in G. Then*

$$\Big|\pi_G(\sigma_u = 1 \mid \sigma_v = 1) - \pi_G(\sigma_u = 1 \mid \sigma_v = 2)\Big| \leq \frac{1}{n^{10}} + \sum_{\ell=1}^{L} P_\ell(G,u,v)\Big(\frac{1-B}{1+B}\Big)^\ell$$

where $P_\ell(G,u,v)$ is the number of paths with ℓ vertices in G that connect u and v.

6.2 Average growth of bichromatic components in the Potts distribution

To utilise the algorithm of Theorem 24 for our Potts sampler, we will need to bound the average growth of bichromatic classes in a typical Potts configuration on a random regular graph. Our key lemma to achieve this will bound the probability that a path is bichromatic[5] when the parameter B is in uniqueness, provided that the local neighbourhood around the path (in the sense of Definition 14) has a tree-like structure. The following lemma quantifies this probability bound and is proved in Section 8 of the full version. The proof technique resembles that of Lemma 18: uniqueness on the tree guarantees that, on the graph $G \backslash P$, the colour of most vertices on the path P is roughly uniformly distributed on $[q]$ and therefore, when we attach the edges of the path P back, the hanging trees can essentially be ignored and we can just focus on the edges of the path.

▶ **Lemma 31.** *Let $\Delta, q \geq 3$ be integers, and $B \in (0,1)$ be in the uniqueness regime of the $(\Delta-1)$-ary tree with $B \neq (\Delta - q)/\Delta$. Then, for any $\epsilon' > 0$, there exists a positive constant $K < \frac{1+B}{B+q-1} + \epsilon'$ and $\epsilon > 0$ such that the following holds for all sufficiently large integers ℓ and h.*

Let G be a graph of maximum degree Δ and P be a path with ℓ vertices whose h-graph-neighbourhood contains $(1-\epsilon)\ell$ isolated tree components. Let μ_G be the Potts measure on G with parameter B. Then, $\mu_G(\text{path } P \text{ is bichromatic}) \leq K^\ell$.

Recall that for a random Δ-regular graph G, paths do have the tree-like structure of Lemma 31 (cf. Lemma 15), and hence we can aggregate over all paths emanating from an arbitrary vertex (roughly $(\Delta-1)^\ell$ of them) and get a bound of roughly $(\Delta-1)K < \frac{(\Delta-1)(1+B)}{B+q-1}$ for the average growth of bichromatic components in a typical configuration σ. This will allow us to use the ReSample subroutine from Section 6.3.

6.3 The resampling subroutine

In Section 3, we discussed that key to the algorithm for the antiferromagnetic case is the resampling subroutine which is invoked when we add a new edge. Here, we give more details about this subroutine and highlight the crucial steps in its analysis.

Let us first abstract somewhat the setting. In particular, let $G = (V, E)$ be a graph and suppose that u, v are vertices in G such that $\{u, v\} \notin E$. Consider the graph $G' = (V, E')$ obtained from G by adding the edge $\{u, v\}$. Given σ distributed according to μ_G, our goal is to produce σ' distributed according to $\mu_{G'}$, perhaps with some small error.

[5] Let $G = (V, E)$ be a graph and $\sigma : V \to [q]$. We call a path P bichromatic under σ if there exist colours $c_1, c_2 \in [q]$ such that every vertex u of P satisfies $\sigma_u \in \{c_1, c_2\}$.

We consider the following procedure. If $\sigma_u \neq \sigma_v$, we set $\sigma' = \sigma$. Otherwise, we flip a coin with heads probability $\frac{qB}{B+q-1}$. If the coin comes up heads, we set $\sigma' = \sigma$. Otherwise, suppose that $\sigma_u = \sigma_v = c$ for some colour $c \in [q]$. Then, we choose uniformly at random a bichromatic class U containing u and v, i.e., we pick uniformly at random a colour $c' \in [q] \backslash c$ and set $U = \sigma^{-1}(c, c')$. Then, we resample the colours on U using a sample from the distribution $\pi^{c,c'}_{G[U],u,v}$ (recall that the latter distribution is the Ising distribution with states $\{c, c'\}$ on the subgraph $G[U]$, conditioned on u taking the colour c and v taking the colour c').

Our key lemma shows that the procedure produces σ' whose distribution $\nu_{\sigma'}$ is close to $\mu_{G'}$, provided that the "average correlation" between u and v in a random bichromatic class containing them is small. To make this precise, for a set $U \subseteq V$ such that $u, v \in U$, let

$$\mathsf{Corr}_G(U, u, v) = \Big| \frac{\pi_{G[U]}\big(\eta_u = 1, \eta_v = 1\big)}{\pi_{G[U]}\big(\eta_u = 1, \eta_v = 2\big)} - 1 \Big|.$$

i.e., $\mathsf{Corr}_G(U, u, v)$ measures the correlation between u and v in the Ising distribution with parameter B on the subgraph $G[U]$. For a configuration σ, we let $U_\sigma \subseteq V$ be a bichromatic class under σ which contains u and v, chosen uniformly at random among the set of all such classes if there is more than one. Then, the term "average correlation" refers to $\mathbf{E}[\mathsf{Corr}_G(U_\sigma, u, v)]$ where the expectation is over the choice of $\sigma \sim \mu_G$ and the choice of the bichromatic class U_σ containing u and v under σ. If this quantity is small, we show in Lemma 29 of the full version that the distance between the distributions $\nu_{\sigma'}$ and $\mu_{G'}$ is also small.

At this stage, there are two points we need to address. First, the procedure assumed that sampling from the distribution $\pi^{c,c'}_{G[U],u,v}$ can be done efficiently which is not always the case. The way to rectify efficiency is the following: once we are given σ, we first check whether all bichromatic classes in σ have average growth less than $b = (\Delta - 1)K$ up to depth $L = \lceil M \log n \rceil$, where K is the constant in Lemma 31 and M is a sufficiently large constant. Using Lemma 31, we can show that this will be the case with very high probability and therefore we can use the algorithm of Theorem 24 to sample from the distribution $\pi^{c,c'}_{G[U],u,v}$; since the sampler is approximate, this introduces a small sampling error ($\leq 1/n^{10}$), which can be safely ignored. (In the unlikely event where σ contains a bichromatic class with average growth larger than b, we can just set σ' to be a configuration chosen uniformly at random). A detailed description of the algorithm can be found in Figures 3 and 4 of the full version.

The second point we need to address is how to bound the complicated-looking quantity $\mathbf{E}[\mathsf{Corr}_G(U_\sigma, u, v)]$ which is relevant for the accuracy of the algorithm. To do this, we first show that, for any U containing u and v, the quantity $\mathsf{Corr}_G(U, u, v)$ is within a constant factor from the quantity

$$\widehat{\mathsf{Corr}}_G(U, u, v) := \big| \pi_{G[U]}\big(\eta_u = 1 \mid \eta_v = 1\big) - \pi_{G[U]}\big(\eta_u = 1 \mid \eta_v = 2\big) \big|.$$

We can bound the latter quantity by Lemma 25, using the number of paths connecting u and v in $G[U]$. In particular, for $\sigma \sim \mu_G$, we can upper bound the (random variable) $\widehat{\mathsf{Corr}}_G(U_\sigma, u, v)$ by a sum, ranging over paths P of logarithmic length, of indicator functions that P is bichromatic. Then, Lemma 31 allows us to upper bound the expectation of this sum. We thus obtain an upper bound on $\mathbf{E}[\mathsf{Corr}_G(U_\sigma, u, v)]$ by a weighted sum over the paths P connecting u and v in the graph G.

The proof of Theorem 32 in the full version gives the formal details behind these arguments.

References

1 K. S. Alexander. Mixing properties and exponential decay for lattice systems in finite volumes. *Ann. Probab.*, 32(1A):441–487, 2004.

2 V. Beffara and H. Duminil-Copin. The self-dual point of the two-dimensional random-cluster model is critical for $q \geq 1$. *Probability Theory and Related Fields*, 153(3):511–542, 2012.

3 M. Bordewich, C. Greenhill, and V. Patel. Mixing of the Glauber dynamics for the ferromagnetic Potts model. *Random Structures & Algorithms*, 48(1):21–52, 2016.

4 G. R. Brightwell and P. Winkler. Random colorings of a Cayley tree. *Contemporary combinatorics*, 10:247–276, 2002.

5 A. Dembo, A. Montanari, A. Sly, and N. Sun. The replica symmetric solution for Potts models on d-regular graphs. *Communications in Mathematical Physics*, 327(2):551–575, 2014.

6 C. Efthymiou. A simple algorithm for random colouring $G(n, d/n)$ using $(2 + \epsilon)d$ colours. In *Proceedings of the Twenty-third Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '12, pages 272–280, 2012.

7 C. Efthymiou. A simple algorithm for sampling colorings of $G(n, d/n)$ up to the Gibbs uniqueness threshold. *SIAM Journal on Computing*, 45(6):2087–2116, 2016.

8 C. Efthymiou, T. P. Hayes, D. Štefankovič, and E. Vigoda. Sampling random colorings of sparse random graphs. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '18, pages 1759–1771, 2018.

9 A. Galanis, L. A. Goldberg, and K. Yang. Uniqueness of the 3-state antiferromagnetic Potts model on the tree. *arXiv/1804.03514*, 2018.

10 A. Galanis, D. Štefankovič, and E. Vigoda. Inapproximability for antiferromagnetic spin systems in the tree nonuniqueness region. *J. ACM*, 62(6):50:1–50:60, 2015.

11 A. Galanis, D. Štefankovič, E. Vigoda, and L. Yang. Ferromagnetic Potts model: Refined #BIS-hardness and related results. *SIAM Journal on Computing*, 45(6):2004–2065, 2016.

12 A. Gerschenfeld and A. Montanari. Reconstruction for models on random graphs. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '07, pages 194–204, 2007.

13 G. Grimmett. *The Random–Cluster Model*. Springer, 2006.

14 O. Häggström. The random-cluster model on a homogeneous tree. *Probability Theory and Related Fields*, 104(2):231–253, 1996.

15 J. Jonasson. The random cluster model on a general graph and a phase transition characterization of nonamenability. *Stochastic Processes and their Applications*, 79(2):335–354, 1999.

16 J. Jonasson. Uniqueness of uniform random colorings of regular trees. *Statistics & Probability Letters*, 57(3):243–248, 2002.

17 F. Martinelli and E. Olivieri. Approach to equilibrium of Glauber dynamics in the one phase region. I. The attractive case. *Comm. Math. Phys.*, 161(3):447–486, 1994.

18 F. Martinelli and E. Olivieri. Approach to equilibrium of Glauber dynamics in the one phase region. II. The general case. *Communications in Mathematical Physics*, 161(3):487–514, 1994.

19 F. Martinelli, E. Olivieri, and R. H. Schonmann. For 2-D lattice spin systems weak mixing implies strong mixing. *Communications in Mathematical Physics*, 165(1):33–47, 1994.

20 M. Mézard and A. Montanari. *Information, Physics, and Computation*. Oxford University Press, 2009.

21 E. Mossel and A. Sly. Rapid mixing of Gibbs sampling on graphs that are sparse on average. *Random Structures & Algorithms*, 35(2):250–270, 2009.

22 E. Mossel and A. Sly. Exact thresholds for Ising–Gibbs samplers on general graphs. *Ann. Probab.*, 41(1):294–328, 2013.

23 E. Mossel, D. Weitz, and N. Wormald. On the hardness of sampling independent sets beyond the tree threshold. *Probability Theory and Related Fields*, 143(3):401–439, 2009.

24 A. Sinclair, P. Srivastava, D. Štefankovič, and Y. Yin. Spatial mixing and the connective constant: optimal bounds. *Probability Theory and Related Fields*, 168(1):153–197, 2017.

25 A. Sinclair, P. Srivastava, and M. Thurley. Approximation algorithms for two-state anti-ferromagnetic spin systems on bounded degree graphs. *Journal of Statistical Physics*, 155(4):666–686, 2014.

26 A. Sinclair, P. Srivastava, and Y. Yin. Spatial mixing and approximation algorithms for graphs with bounded connective constant. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013*, pages 300–309, 2013.

27 L. E. Thomas. Bound on the mass gap for finite volume stochastic Ising models at low temperature. *Communications in Mathematical Physics*, 126(1):1–11, 1989.

28 Y. Yin and C. Zhang. Sampling in Potts model on sparse random graphs. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2016*, pages 47:1–47:22, 2016.

29 J. Zhang, H. Liang, and F. Bai. Approximating partition functions of the two-state spin system. *Information Processing Letters*, 111(14):702–710, 2011.

Polar Codes with Exponentially Small Error at Finite Block Length

Jarosław Błasiok[1]
Harvard John A. Paulson School of Engineering and Applied Sciences,
Harvard University, 33 Oxford Street, Cambridge, MA 02138, USA
jblasiok@g.harvard.edu

Venkatesan Guruswami[2]
Computer Science Department,
Carnegie Mellon University, Pittsburgh, PA 15213, USA.
venkatg@cs.cmu.edu

Madhu Sudan[3]
Harvard John A. Paulson School of Engineering and Applied Sciences,
Harvard University, 33 Oxford Street, Cambridge, MA 02138, USA.
madhu@cs.harvard.edu

Abstract

We show that the entire class of polar codes (up to a natural necessary condition) converge to capacity at block lengths polynomial in the gap to capacity, while *simultaneously* achieving failure probabilities that are exponentially small in the block length (i.e., decoding fails with probability $\exp(-N^{\Omega(1)})$ for codes of length N). Previously this combination was known only for one specific family within the class of polar codes, whereas we establish this whenever the polar code exhibits a condition necessary for any polarization.

Our results adapt and strengthen a local analysis of polar codes due to the authors with Nakkiran and Rudra [Proc. STOC 2018]. Their analysis related the time-local behavior of a martingale to its global convergence, and this allowed them to prove that the broad class of polar codes converge to capacity at polynomial block lengths. Their analysis easily adapts to show exponentially small failure probabilities, provided the associated martingale, the "Arıkan martingale", exhibits a corresponding strong local effect. The main contribution of this work is a much stronger local analysis of the Arıkan martingale. This leads to the general result claimed above.

In addition to our general result, we also show, for the first time, polar codes that achieve failure probability $\exp(-N^{\beta})$ for any $\beta < 1$ while converging to capacity at block length polynomial in the gap to capacity. Finally we also show that the "local" approach can be combined with *any* analysis of failure probability of an arbitrary polar code to get essentially the same failure probability while achieving block length polynomial in the gap to capacity.

2012 ACM Subject Classification Mathematics of computing → Coding theory

Keywords and phrases Polar codes, error exponent, rate of polarization

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.34

[1] Supported by ONR grant N00014-15-1-2388.
[2] This work was done when the author was visiting the Center of Mathematical Sciences and Applications, Harvard University, Cambridge, MA 02138. Research supported in part by NSF grants CCF-1422045 and CCF-1563742.
[3] Work supported in part by a Simons Investigator Award and NSF Awards CCF 1565641 and CCF 1715187.

1 Introduction

Ever since their discovery [1] polar codes have been a subject of vast interest, both for their theoretical and practical significance. Theoretical interest in them arises from two desirable features that they exhibit: (1) They give codes of length N (for infinitely many N) along with efficient decoding algorithms that correct channel errors with all but exponentially (i.e., $\exp(-N^{\Omega(1)})$) small failure probability. (2) They also converge to capacity extremely fast - i.e., at block length N which is only polynomial in the inverse of the "gap to capacity". The former effect is known to hold in general, i.e., for the entire class of polar codes (up to a minimal and natural necessary condition). The latter was shown to hold in the same generality only recently [2] — previous works [5, 6, 4] were only able to establish it for one specific construction of polar codes. And while the early works were able to show effects (1) and (2) simultaneously for this construction, the other polar codes were not known to be able to show both features simultaneously.

The main goal of this paper is to remedy this weakness. We show roughly that the techniques of [2] can be strengthened to achieve both effects simultaneously for the entire broad class of polar codes. In addition to the generality of the result this also leads to quantitative improvements on the error-exponent at polynomially small block lengths. We elaborate on these further after some background.

1.1 Background

In the theory of Shannon, a memoryless channel is given by a probabilistic map from an input alphabet (a finite field $\mathbb{F}_q$ in this paper) to an output alphabet (an abstract set $\mathcal{Y}$ here). A family of codes $C_N : \mathbb{F}_q^{k_N} \to \mathbb{F}_q^N$ along with decoding algorithm $D_N : \mathcal{Y}^N \to \mathbb{F}_q^{k_N}$ achieves *rate* R if $\lim_{N\to\infty} k_N/N \geq R$. It is said to achieve failure probability $\mathrm{err}(N)$ if $\Pr_{M\in\mathbb{F}_q^{k_N}}[D_N(C_N(M) \neq M] \leq \mathrm{err}(N)$ for every N. Shannon's celebrated theorem associates a capacity C with every channel such that transmission at rate higher than capacity will have constant failure probability, whereas for every $R < C$, for every sufficiently large N, there exist codes of rate R with failure probability $\exp(-\Omega(N))$. The quantity $\varepsilon \triangleq C - R$ is called the "gap to capacity". The relationship between the block length N, the gap to capacity ε and the failure probability $\mathrm{err}(N)$ are the central quantities of interest to this paper.

The specific family of codes we consider in this paper are "polar codes" introduced by Arıkan [1]. These codes are a broad class of (infinite families of) codes, one family for every matrix $M \in \mathbb{F}_q^{k\times k}$ and symmetric channel. The t-th code in the sequence has length k^t, and is given by (affine shift) of some subset of rows of $M^{\otimes t}$. It is well known that under a simple necessary condition on M (that we call mixing), these codes achieve exponentially small failure probability in a weak sense: Specifically for every symmetric channel, for every mixing M, there exists a $\beta > 0$ such that for every $\varepsilon > 0$ there exists a N_0 such that every code in the family of length $N \geq N_0$ has at most ε gap to capacity and achieves failure probability at most $\exp(-N^\beta)$. Indeed by picking M carefully one could achieve β arbitrarily close to 1 (though this approach can not yield $\beta = 1$), and moreover for a given matrix M, the range of achievable β can be explicitly computed from simple combinatorial properties of this matrix [7]. However note that these analyses did not provide explicit relationship between ε and N_0.

It was more recently shown [5, 6, 4] that there exists an M (specifically $M = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$) such that the associated code achieves exponentially small failure probability even at polynomially small block lengths — i.e., when $N_0 = \mathrm{poly}(1/\varepsilon)$. The β associated with this result is bounded well away from 1. But till last year no other code (for any other matrix M)

was even known to achieve failure probability going to zero for polynomially small block lengths. This was remedied in part by a previous work of the authors with Nakkiran and Rudra [2] where they showed that for every mixing matrix M and every symmetric channel the associated code converges at block length growing polynomially with gap to capacity, however their failure probability analysis only yielded $\mathrm{err}(N) \leq 1/\operatorname{poly}(N)$. Their work forms the starting point of this work.

1.2 Our results

Our results show that it is possible to combine the general analyses for "polynomial convergence of block length in gap to capacity" (from [2]) with any strong analysis of the failure probability. Specifically we show the following:

1. For every mixing matrix M and symmetric channel the associated family of polar codes yield exponentially small decoding failure at block lengths polynomial in the gap to capacity.
2. While the result in Part (1) is general the resulting β may not be optimal. We complement this with a result showing that for every $\beta < 1$ there exist polar codes, that get close to capacity at polynomial block length with decoding failure probability being $\exp(-N^{\beta})$. We note that no previous analysis yielded such quantitatively strong bounds on any family of polar codes with polynomial block length.
3. Finally we show that convergence to capacity at polynomial block length comes with almost no price in the failure probability. We show this by show that if any polar code achieves capacity (even if at very large block lengths) with failure probability $\exp(-N^{\beta})$, then for every $\beta' < \beta$ it achieves capacity with failure probability $\exp(-N^{\beta'})$ where the block length is a polynomial $p_{\beta,\beta'}(1/\varepsilon)$.

While the third result subsumes the previous two (when combined with known results in the literature), we include the first two to show that it is possible to prove strong results about failure probabilities $\exp(-N^{\beta})$ with blocklength polynomial in the gap to capacity, entirely within the local polarization framework developed in [2] and here — without appealing to previous analyses. In fact the proofs of those two are quite simple (given the work of [2]).

On the other hand, for given matrix M, the optimal exponent β was exactly characterized in terms of explicit combinatorial properties of matrix M — but with potentially very large blocklengths [7]. The third result of our paper automatically lifts this theorem to the setting where blocklength is polynomial in the gap to capacity — given matrix M one can compute the "correct" exponent β as in [7], and essentially the same exponent is achievable already within polynomial blocklength, whereas no larger exponent is achievable, regardless of how large blocklength one takes.

1.3 Techniques

We now turn to the central ingredient in our analyses of polar codes which we inherit from [2], namely the "local" analysis of $[0,1]$-martingales. It is well-known that the analysis of polar codes can be tied to the analysis of an associated martingale, called the Arıkan martingale in [2]. Specifically given a channel and a matrix M one can design a martingale $X_0, X_1, \ldots, X_t, \ldots$ with $X_t \in [0,1]$, such that the performance of the code of length k^t depends on the behavior of the random variable X_t. Specifically to achieve ε gap to capacity with failure probability $\rho = \mathrm{err}(N)$, the associated martingale should satisfy $\Pr[X_t \in (\rho/N, 1-\varepsilon/2)] \leq \varepsilon/2$. Considering the fact that we want the failure to be

exponentially small in N and ε to be inverse polynomially small in N and noting $N = k^t$, this requires us to prove that $\Pr[X_t \in (\exp(-\exp(O(t))), 1 - \exp(-\Omega(t))] \leq \exp(-\Omega(t))$.

Usual proofs of this property typically track many aspects of the distribution of X_t, whereas a "local" analysis simply reasons about the distribution of X_t conditioned on X_{t-1}. For the Arıkan martingale (as for many other natural martingales) this one-step evolution is much easier to describe than the cumulative effects of t-steps. In [2] a simple local property, called "local polarization", of this one-step evolution was described (enforcing that the random variable has enough variance if it is not close to the boundary $\{0, 1\}$ and that it gets sucked to the boundary when it is close). It was then shown that local polarization leads to global polarization, though only for $\rho = 1/\operatorname{poly}(N)$ — specifically they showed that $\Pr[X_t \in (1/\operatorname{poly}(N), 1 - \varepsilon/2)] \leq \varepsilon/2$.

It is easy to modify the definition of local polarization slightly to get a stronger definition that would imply the desired convergence even for $\rho(N) = \exp(-N^{\Omega(1)})$. Indeed we do so, calling it "exponential local polarization" of a martingale, and show that this stronger local polarization leads to exponentially small failure probabilities.

The crux of this paper is in showing that the Arıkan martingale exhibits exponential local polarization. For readers familiar with the technical aspects, this might even be surprising. In fact the most well-studied Arıkan martingale, the one associated with the binary symmetric channel and the matrix $M = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$ is not exponentially locally polarizing. We get around this seemingly forbidding barrier by showing that the martingale associated with $M^{\otimes 2}$ (the tensor-product of M with itself) is exponentially locally polarizing, and this is almost as good for us. (Instead of reasoning about the martingale $X_0, X_1, X_2, \ldots$, this allows us to reason about $X_0, X_2, X_4, \ldots$ which is sufficient for us.) Combined with some general reductions as in [2] this allows us to show that for every symmetric channel and every mixing matrix, the associated martingale is exponentially locally polarizing and this yields our first main result above.

To get failure probability $\exp(-N^\beta)$ for $\beta \to 1$ we show that if the matrix M contains the parity check matrix of a code of sufficiently high distance then the Arıkan martingale associated with M shows exponential local polarization over any symmetric channel, and in turn this leads to codes whose failure probability is $\exp(-N^\beta)$ for $\beta \to 1$.

Finally we turn to our last result showing that any matrix producing codes with failure probability $\exp(-N^\beta)$ (but not necessarily for $N = \operatorname{poly}(1/\varepsilon)$) also gets failure probability $\exp(-N^{\beta'})$ for $N \geq p_{\beta,\beta'}(1/\varepsilon)$ for some polynomial $p_{\beta,\beta'}$, and any $\beta' < \beta$. This result is obtained by showing that if M achieves exponentially small error, then for some large $t_0 = t_0(\beta, \beta')$, the matrix $M^{\otimes t_0}$ contains the parity check matrix of a high-distance code, with distance high enough to imply that its failure probability is $\exp(-N^{\beta'})$.

2 Main Definitions and Results

2.1 Martingales and Polarization

In this section we let $X_0, X_1, X_2, \ldots$ be a $[0, 1]$-bounded martingale, i.e., $X_t \in [0, 1]$ for all t and for every $x_0, \ldots, x_t$, $\mathbb{E}[X_{t+1} | X_0 = x_0, \cdots, X_t = x_t] = x_t$.

We say that a martingale has exponentially strong polarization if the probability that X_t is not close (as a function of t) to the boundary $\{0, 1\}$ is exponentially small in t. Formally

▶ **Definition 1** (Exponentially Strong Polarization)**.** We say that X_t has Λ-exponentially strong polarization if for every $0 < \gamma < 1$ there exist constants $\alpha < \infty$ and $0 < \rho < 1$ such that for every t, $\Pr[X_t \in (2^{-2^{\Lambda \cdot t}}, 1 - \gamma^t)] \leq \alpha \cdot \rho^t$.

Note that this definition is asymmetric — paths of the martingale that converge to zero, have doubly-exponential rate of convergence, whereas those converging to 1 are doing it only exponentially fast.[4] This should be compared with the notion of strong polarization present in [2], namely

▶ **Definition 2** (Strong Polarization)**.** We say that X_t has strong polarization if for every $0 < \gamma < 1$ there exist constants $\alpha < \infty$ and $0 < \rho < 1$ such that for every t, $\Pr[X_t \in (\gamma^t, 1-\gamma^t)] \leq \alpha \cdot \rho^t$.

As in [2] the notion of Exponential Strong Polarization is not a local one but rather depends on the long run behavior of X_t. A notion of local polarization, that only relates the evolution of X_{t+1} from X_t, was defined in [2], and shown to imply strong polarization. Let us recall this definition.

▶ **Definition 3** (Local Polarization)**.** A $[0,1]$-martingale sequence $X_0, \ldots, X_j, \ldots$, is *locally polarizing* if the following conditions hold:

1. **(Variance in the middle):** For every $\tau > 0$, there is a $\theta = \theta(\tau) > 0$ such that for all j, we have: If $X_j \in (\tau, 1-\tau)$ then $\mathbb{E}[(X_{j+1} - X_j)^2 | X_j] \geq \theta$.
2. **(Suction at the ends):** There exists an $\alpha > 0$, such that for all $c < \infty$, there exists a $\tau = \tau(c) > 0$, such that:
   a. If $X_j \leq \tau$ then $\Pr[X_{j+1} \leq X_j/c | X_j] \geq \alpha$.
   b. Similarly, if $1 - X_j \leq \tau$ then $\Pr[(1 - X_{j+1} \leq (1-X_j)/c | X_j] \geq \alpha$.

   We refer to condition (a) above as *Suction at the low end* and condition (b) as *Suction at the high end*.

When we wish to be more explicit, we refer to the sequence as $(\alpha, \tau(\cdot), \theta(\cdot))$-locally polarizing.

With an eye toward showing exponential strong polarization also via a local analysis, we now define a concept of local polarization tailored to exponential polarization.

▶ **Definition 4** (Exponential Local Polarization)**.** We say that X_t has (η, b)-exponential local polarization if it satisfies local polarization, and the following additional property

1. **(Strong suction at the low end):** There exists $\tau > 0$ such that if $X_j \leq \tau$ then $\Pr[X_{j+1} \leq X_j^b | X_j] \geq \eta$.

In the same way as local polarization implies the strong global polarization of a martingale [2, Theorem 1.6], this new stronger local condition implies a stronger global polarization behavior.

▶ **Theorem 5** (Local to Global Exponential Polarization)**.** *Let $\Lambda < \eta \log_2 b$. Then if a $[0,1]$-bounded martingale $X_0, X_1, X_2, \ldots$ satisfies (η, b)-exponential local polarization then it also satisfies Λ-exponentially strong polarization.*

The proof of this theorem follows the same outline as the proof of Theorem 1.6 in [2], and we present it in Section A.

[4] It turns out that for the polar coding application, the behavior of the martingale at the lower end is important as it governs the decoding error probability, whereas behavior of the martingale near the upper end is not that important. The probability that the martingale doesn't polarize corresponds to the gap to capacity.

2.2 Matrix Polarization

In this section we relate statements about the local polarization of the Arıkan martingale associated with some matrix M (and some channel) to structural properties of M itself. The formal definition of the Arıkan martingale is included for completeness in Appendix B, but will not be used in this paper.

We first rewrite the (technical) condition of the Arıkan martingale associated with M being exponentially locally polarizing in more direct terms. This leads us to the following definition.

▶ **Definition 6** (Exponential polarization of matrix)**.** We say that a matrix $M \in \mathbb{F}_q^{k\times k}$ satisfies (η, b)-exponential polarization, if there exist some $\tau > 0$, such that for any $\delta < \tau$ and for any random sequence $(U_1, A_1), \dots (U_k, A_k)$, where $(U_i, A_i) \in \mathbb{F}_q$ are i.i.d., and satisfy $\overline{H}(U_i|A_i) \leq \delta$, we have

$$\overline{H}((UM)_j|(UM)_{<j}, A) \leq \delta^b$$

for at least η fraction of indices $j \in [k]$.

In the above definition and throughout the paper $\overline{H}$ refers to normalized entropy, i.e. $\overline{H}(X|A) := \frac{1}{\log_2 q} H(X|A)$, so that $\overline{H}(X|A) \in [0, 1]$, and $U = (U_1, \dots U_k)$, similarly $A = (A_1, \dots A_k)$. Moreover, for a vector $V \in \mathbb{F}^k$, and $j \leq k$, by $V_{<j}$ we denote a vector in $\mathbb{F}^{j-1}$ with coordinates $(V_1, \dots V_{j-1})$.

The following lemma explicitly asserts that matrix polarization implies martingale polarization (as claimed).

▶ **Lemma 7.** *If matrix M satisfies (η, b)-exponential polarization, then Arıkan martingale associated with M is (η, b)-exponentially locally polarizing.*

The proof of the above lemma is very similar to the proof of Theorem 1.10 in [2] — with definitions of Arıkan martingale and exponential polarization of matrix in hand this proof is routine, although somewhat tedious and notationally heavy. We postpone this proof to the full version of this paper.

In the light of the above, and in context of Theorem 5, we have reduced the problem of showing (global) exponentially strong polarization of Arıkan martingale, to understanding parameters for exponential polarization of specific matrices, based on the structural propertues of these matrices.

In this paper we provide three results of this form. The first of our results considers mixing matrices and analyzes their local polarization. We recall the definition of a mixing matrix.

▶ **Definition 8** (Mixing matrix)**.** 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 rows of M, the resulting matrix is not upper-triangular.

It is well known that if a matrix is not mixing then the associated martingale does not polarize at all (and the corresponding martingale satisfies $X_t = X_{t-1}$ for every t). In contrast if the matrix M is mixing, our first lemma shows that $M^{\otimes 2}$ (the tensor-product of M with itself) is exponentially polarizing.

▶ **Lemma 9.** *For every mixing matrix $M \in \mathbb{F}_q^{k\times k}$ and every $\varepsilon > 0$, matrix $M^{\otimes 2}$ satisfies $(\frac{1}{k^2}, 2 - \varepsilon)$-exponential polarization.*

This translates immediately to our first main theorem stated in Section 2.3.

Our second structural result on matrix polarization shows that matrices that contain the parity check matrix of a high distance code lead to very strong exponential polarization parameters.

▶ **Lemma 10.** *If a mixing matrix M is decomposed as $M = [M_0|M_1]$, where $M_0 \in \mathbb{F}_q^{k\times(1-\eta)k}$ is such that $\ker M_0^T$ is a linear code of distance larger than $2b$, then matrix M satisfies $(\eta, b-\varepsilon)$-exponential polarization for every $\varepsilon > 0$.*

By using standard results on existence of codes with good distance, we get as an immediate corollary that there exist matrices with almost optimal exponential polarization parameters.

▶ **Corollary 11.** *For every ε and every prime field $\mathbb{F}_q$, there exist k, and matrix $M \in \mathbb{F}_q^{k\times k}$, such that matrix M satisfies $(1-\varepsilon, k^{1-\varepsilon})$ exponential polarization.*

Proof. Consider a parity check matrix M_0 of a BCH code with distance $2k^{1-\varepsilon}$. We can achieve this with a matrix $M_0 \in \mathbb{F}_q^{k\times k_0}$, where $k_0 = \mathcal{O}(k^{1-\varepsilon}\log k)$. Hence, as soon as $k > \Omega(2^{\varepsilon^{-1}\log\varepsilon^{-1}})$, we have $k_0 < \varepsilon k$. We can now complete M_0 to a mixing matrix. ◀

It is worth noting, that by the same argument and standard results on the distance of random linear codes, a random matrix $M \in \mathbb{F}_q^{k\times k}$ with high probability satisfies a $(1-\varepsilon, k^{1-\varepsilon})$ local polarization, with $\varepsilon \to 0$ as $k \to \infty$.

By the whole chain of reductions discussed above, Corollary 11 implies that for any ε there exist polar codes with decoding failure probability $\exp(-N^{1-\varepsilon})$, where the blocklength N depends polynomially in the desired gap to capacity. Moreover, those codes are ubiquitous — polar codes arising from a large random matrix will usually have this property.

Our final structural result is morally a "converse" to the above: It shows that if a matrix M leads to a polar code with exponentially small failure probability then some high tensor power $N = M^{\otimes t}$ of M contains the parity check matrix of a high distance code. In fact more generally if a matrix $P \in \mathbb{F}_q^{k\times s}$ is the parity check matrix of a code which has a decoding algorithm that corrects errors from a q-symmetric channel with failure probability $\exp(-k^\beta)$ then this code has high distance.

▶ **Definition 12.** For any finite field $\mathbb{F}_q$ we will denote by $B_q(\varepsilon)$ distribution on $\mathbb{F}_q$ such that for $Z \sim B_q(\varepsilon)$ we have $\Pr(Z=0) = 1-\varepsilon$, and $\Pr(Z=k) = \frac{\varepsilon}{q-1}$ for any $k \neq 0$.

▶ **Lemma 13.** *Consider a matrix $P \in \mathbb{F}_q^{k\times s}$ and arbitrary decoding algorithm $\mathrm{Dec} : \mathbb{F}_q^s \to \mathbb{F}_q^k$, such that for independent random variables $U_1, \dots U_i \sim B_q(\varepsilon)$ with $\varepsilon < \frac{1}{2}$, we have $\Pr(\mathrm{Dec}(UP) \neq U) < \exp(-k^\gamma)$. Then $\ker P$ is a code of distance $k^\gamma \log^{-1}(q/\varepsilon)$.*

2.3 Implications for polar codes

We start this section by including the definition of symmetric channel — all our results about polar codes show that we can achieve capacity for those channels.

▶ **Definition 14** (Symmetric memoryless channel)**.** A q-ary *symmetric memoryless channel* is any probabilistic function $\mathcal{C} : \mathbb{F}_q \to \mathcal{Y}$, such that for every $\alpha, \beta \in \mathbb{F}_q$ there is a bijection $\sigma : \mathcal{Y} \to \mathcal{Y}$ such that for every $y \in \mathcal{Y}$ it is the case that $\mathcal{C}_{Y=y|\alpha} = \mathcal{C}_{Y=\sigma(y)|\beta}$, and moreover for any pair $y_1, y_2 \in \mathcal{Y}$, we have $\sum_{x\in\mathbb{F}_q} C_{Y=y_1|x} = \sum_{x\in\mathbb{F}_q} C_{Y=y_2|x}$ (see, for example, [3, Section 7.2]).

Such probabilistic function yields a probabilistic function $\mathcal{C} : \mathbb{F}_q^N \to \mathcal{Y}^N$, by acting independently on each coordinate.

We will now recall the following theorem which shows that if the Arıkan martingale polarizes then a corresponding code achieves capacity with small failure probability.

▶ **Theorem 15** (Implied by Arıkan [1]). *Let $\mathcal{C}$ be a q-ary symmetric memoryless channel and let $M \in \mathbb{F}_q^{k\times k}$ be an invertible matrix. If the Arıkan martingale associated with $(M, \mathcal{C})$ is Λ-exponentially strongly polarizing then there is a polynomial p such that for every $\varepsilon > 0$ and every $N = k^t \geq p(1/\varepsilon)$, there is a code $C \subseteq \mathbb{F}_q^N$ of dimension at least $(\mathrm{Capacity}(\mathcal{C}) - \varepsilon) \cdot n$ such that C is an affine code generated by the restriction of $(M^{-1})^{\otimes t}$ to a subset of its rows and an affine shift. Moreover there is a decoding algorithm for these codes that has failure probability bounded by $\exp(-N^{\Lambda/\log_2 k})$, and running time $\mathcal{O}(N \log N)$. The running time of accompanying encoding algorithm is also $\mathcal{O}(N \log N)$.*

We omit the proof of this theorem, which is identical to Theorem 1.7 in [2] except for minor modifications to incorporate the exponential polarization/failure probability.

Armed with this theorem, we can now convert the structural results asserted in the previous section into convergence and failure probability of polar codes.

▶ **Theorem 16.** *For every prime q, every mixing matrix $M \in \mathbb{F}_q^{k\times k}$, every symmetric memoryless channel $\mathcal{C}$ over $\mathbb{F}_q$, there is a polynomial p and $\beta > 0$ such that for every $\varepsilon > 0$ and every $N = k^t \geq p(1/\varepsilon)$, there is an affine code C, that is generated by the rows of $(M^{-1})^{(\otimes t)}$ and an affine shift, with the property that the rate of C is at least $\mathrm{Capacity}(\mathcal{C}) - \varepsilon$, and C can be encoded and decoded in time $\mathcal{O}(N \log N)$ and failure probability at most $\exp(-N^\beta)$.*

Proof. Follows by composing Lemma 9, Lemma 7, Theorem 5, and 15. ◀

▶ **Theorem 17.** *For every prime q, every symmetric memoryless channel $\mathcal{C}$ over $\mathbb{F}_q$, and every $\beta < 1$, there exists k, a mixing matrix $M \in \mathbb{F}_q^{k\times k}$, and a polynomial p such that for every $\varepsilon > 0$ and every $N = k^t \geq p(1/\varepsilon)$, there is an affine code C, that is generated by the rows of $(M^{-1})^{(\otimes t)}$ and an affine shift, with the property that the rate of C is at least $\mathrm{Capacity}(\mathcal{C}) - \varepsilon$, and C can be encoded and decoded in time $\mathcal{O}(N \log N)$ and failure probability at most $\exp(-N^\beta)$.*

Proof. Follows by composing Corollary 11, Lemma 7, Theorem 5, and 15. ◀

▶ **Theorem 18.** *Suppose $M \in \mathbb{F}_q^{k\times k}$ and $\beta > 0$ satisfy the condition that for every memoryless symmetric additive channel[5] $\mathcal{C}$ and for every $\varepsilon > 0$, for sufficiently large $n = k^s$, there is an affine code C of length n generated by the rows of $(M^{-1})^{(\otimes s)}$ of rate at least $\mathrm{Capacity}(\mathcal{C}) - \varepsilon$ such that C can be decoded with failure probability at most $\exp(-n^\beta)$.*

Then, for every $\beta' < \beta$ and every symmetric channel $\mathcal{C}'$, there is a polynomial p such that for every $\varepsilon > 0$ and every $N = k^t \geq p(1/\varepsilon)$ there is an affine code C, that is generated by the rows of $(M^{-1})^{(\otimes t)}$ and an affine shift, with the property that the rate of C is at least $\mathrm{Capacity}(\mathcal{C}') - \varepsilon$, and C can be encoded and decoded in time $\mathcal{O}(N \log N)$ and failure probability at most $\exp(-N^{\beta'})$.

We prove this theorem in Section 4.

Note that in this theorem, we assume that M achieves failure probabilities $\exp(-N^\beta)$ for *additive* channels (which is only a subclass of all symmetric channels), to conclude that it

[5] An additive symmetric channel is a special case of symmetric channels, where the output is the sum of the input with an "error" generated independently of the input.

achieves failure probability $\exp(-N^{\beta'})$ for *all* symmetric channels. This is potentially useful, as proving good properties of polar codes for additive channels is often simpler — in this setting there is a very clean equivalence between coding and linear compression schemes.

This lemma, when combined with Lemma 10 shows that the only way a polar code associated with a matrix M can give exponentially small failure probability $\exp(-N^{\beta})$ is that some tensor of this matrix is *locally exponentially* polarizing and so in particular this matrix also yields exponentially small failure probabilities at block length polynomial in the gap to capacity.

3 Structural analysis of matrices

3.1 Exponential polarization for all mixing matrices

We will first prove that a single specific matrix, namely $\begin{bmatrix} 1 & 0 \\ \alpha & 1 \end{bmatrix}$, after taking second Kronecker power satisfies exponential polarization. In [2] local polarization of any mixing matrix was shown essentially by reducing to this case. Here we make this reduction more explicit, so that it commutes with taking Kronecker product of a matrix with itself. That is, we will later show that for any mixing matrix M exponential polarization of $M^{\otimes 2}$ can be reduced to exponential polarization of $\begin{bmatrix} 1 & 0 \\ \alpha & 1 \end{bmatrix}^{\otimes 2}$.

▶ **Lemma 19.** *Consider* $M = \begin{bmatrix} 1 & 0 \\ \alpha & 1 \end{bmatrix}$ *for nonzero* $\alpha \in \mathbb{F}_q$. *For every* $\varepsilon > 0$ *matrix* $M^{\otimes 2}$ *satisfies* $(\frac{1}{4}, 2-\varepsilon)$ *exponential polarization.*

Proof. Consider arbitrary sequence of i.i.d. random variables $(U_1, A_1), \dots (U_4, A_4)$ with $H(U_i|A_i) = \delta$, as in the definition of exponential polarization. We can explicitly write down matrix $M^{\otimes 2}$ as

$$M^{\otimes 2} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \alpha & 1 & 0 & 0 \\ \alpha & 0 & 1 & 0 \\ \alpha^2 & \alpha & \alpha & 1 \end{bmatrix}.$$

Matrix $M^{\otimes 2}$ has four rows — to achieve $\eta = \frac{1}{4}$ parameter of exponential polarization, we just need to show that there is at least one index i satisfying the inequality as in the definition of exponential polarization (Definition 6). Let us consider vector $U = (U_1, \dots U_4)$ and similarly $A = (A_1, \dots A_4)$. We want to bound

$$\begin{aligned} \overline{H}((UM^{\otimes 2})_4 | (UM^{\otimes 2})_{<4}, A) &= \overline{H}(U_4 | U_1 + \alpha U_2 + \alpha U_3 + \alpha^2 U_4, U_2 + \alpha U_4, U_3 + \alpha U_4, A) \\ &\leq \overline{H}(U_4 | U_2 + \alpha U_4, U_3 + \alpha U_4, A) \end{aligned}$$

By Lemma 31 there exist some function $f : \Sigma \to \mathbb{F}_q$, such that $\Pr(f(A_i) \neq U_i) \leq \delta$. Now, given vector A and $W_2 := \alpha U_4 + U_2, W_3 := \alpha U_4 + U_3$, we can try to predict U_4 as follows: if $W_2 - f(A_2) = W_3 - f(A_3)$ we report $\hat{U}_4 := \alpha^{-1}(W_2 - f(A_2))$. Otherwise, we report $\hat{U}_4 := f(A_4)$.

We want to show that $\Pr(\hat{U}_4 \neq U_4) \leq 3\delta^2$. Indeed, $\hat{U}_4 \neq U_4$ only if at least two of the variables $U_i - f(A_i)$ for $i \in \{2, 3, 4\}$ are non-zero. By symmetry, we have $\Pr(\hat{U}_4 \neq U_4) \leq 3\Pr(U_1 \neq f(A_1) \wedge U_2 \neq f(A_2)) = 3\Pr(U_1 \neq f(A_1))^2 \leq 3\delta^2$.

By Fano's inequality 32, we have $\overline{H}(U_4|U_2+\alpha U_4, U_3+\alpha U_4, A) \leq 6\delta^2(\log \delta^{-1}+\log q+\log 3)$. For any given ε, there exist τ such that if $\delta < \tau$ we have $6(\log \delta^{-1} + \log q + \log 3) \leq \delta^{-\varepsilon}$, hence for those values of δ we have $\overline{H}((UM^{\otimes 2})_4|(UM^{\otimes 2})_{<4}, A) \leq \delta^{2-\varepsilon}$. ◀

We will now proceed to show that exponential polarization for $M^{\otimes 2}$ of any mixing matrix M can be reduced to the theorem above. To this end we define the following containment relation for matrices.

▶ **Definition 20** (Matrix (useful) containment)**.** We say that a matrix $M \in \mathbb{F}_q^{k\times k}$ contains a matrix $R \in \mathbb{F}_q^{m\times m}$, if there exist some $T \in \mathbb{F}_q^{k\times m}$ and a permutation matrix $P \in \mathbb{F}_q^{k\times k}$, such that $PMT = \begin{bmatrix} R \\ 0 \end{bmatrix}$. If moreover the last non-zero row of T is rescaling of the standard basis vector $T_j = \alpha e_m$, we say that containment is R in M is useful and we denote it by $R \sqsubset_u M$. Note that useful containment is *not* a partial order.

The following fact about useful containment will be helpful.

▶ **Claim 21.** *If $R \sqsubset_u M$, then for any upper triangular matrix U with diagonal elements $U_{i,i} = 1$, we also have $R \sqsubset_u MU^{-1}$.*

Proof. Consider matrix T and permutation P as in the definition of useful containment for $R \sqsubset_u M$. We can pick the very same permutation P and matrix $T' = UT$ to witness $R \sqsubset_u MU^{-1}$. All we have to show is that last non-zero row of T' is standard basis vector e_m. Indeed, if j_0 is the last non-zero row of T, and $j > j_0$, rows $(U)_j$ are supported exclusively on elements with indices larger than j_0, hence $(UT)_j = (U)_j T = 0$. On the other hand $(UT)_{j_0} = \sum_i U_{j_0,i}T_i = \sum_{i\geq j_0} U_{j_0,i}T_i = \alpha e_m$, where the last equality follows from the fact that T was useful — that is $T_{j_0} = \alpha e_m$ and $T_i = 0$ for $i > j_0$. ◀

Results of the Lemma 5.5 in [2] can be reintepreted as the following Lemma. We give a full new proof here, as we describe it now in the language of useful containment.

▶ **Lemma 22.** *Every mixing matrix $M \in \mathbb{F}_q^{k\times k}$ contains matrix $H = \begin{bmatrix} 1 & 0 \\ \alpha & 1 \end{bmatrix}$ in a useful way.*

Proof. For any matrix M, there is some permutation matrix P and pair L, U, such that $PM = LU$ where L is lower triangular, and U is upper triangular. Matrix M being mixing is equivalent to the statement that L and U are invertible, and moreover L is not diagonal. As such by Claim 21 it is enough to show that any lower-triangular L, which is not diagonal, contains H in a useful way. Indeed, let s be the last column of L that contains more than a single non-zero entry, and let r to be the last row of non-zero entry in column $L_{\cdot,s}$. Note that column $L_{\cdot,r}$ has single non-zero entry $L_{r,r} = 1$. We will show a matrix $T \in \mathbb{F}_q^{k\times 2}$ as in the definition of useful containment. Let us specify a second column of $T_{\cdot,2} := e_r$. To specify the first column of T we wish to find a linear combination of columns of $L_{1,\cdot}, \ldots, L_{r-1,\cdot}$ such that $\sum_{i\leq r-1} t_i L_{i,\cdot} = \alpha e_s + \alpha e_r$. Then coefficeints t_i can be used as the first column of matrix M. We can set those coefficients to $t_i = -L_{s,i}$ for $i \in [s+1, r-1]$, and $t_s = 1$ — this setting is correct, because columns $L_{i,\cdot}$ for $i \in [s+1, r-1]$ has only one non-zero entry $L_{i,i}$. Now if P is any matrix corresponding to a permutation which maps $s \mapsto 1$ and $r \mapsto 2$, the containemnt $H \sqsubset_u L$ is witnessed by pair P and T. ◀

▶ **Lemma 23.** *If matrix $R \sqsubset_u M$ where $R \in \mathbb{F}_q^{s\times s}$ and $M \in \mathbb{F}_q^{k\times k}$, then $R^{\otimes 2} \sqsubset_u M^{\otimes 2}$.*

Proof. Consider matrix T and permutation P as in the definition of useful containment for $R \sqsubset_u M$. Note that $P^{\otimes 2}M^{\otimes 2}T^{\otimes 2} = (PMT)^{\otimes 2}$. As such, restriction of a matrix $P^{\otimes 2}M^{\otimes 2}T^{\otimes 2}$ to rows corresponding to $[k] \times [k]$ is exactly R, and all remaning rows are zero. We can apply additional permutation matrix $\tilde{P}$ so that those are exactly first k^2 rows of the matrix $\tilde{P}P^{\otimes 2}M^{\otimes 2}T^{\otimes 2}$ give matrix $R^{\otimes 2}$, and the remaining rows are zero. ◀

▶ **Lemma 24.** *If matrix M contains matrix $R = \begin{bmatrix} 1 & 0 \\ \alpha & 1 \end{bmatrix}^{\otimes 2}$ in a useful way, then matrix M satisfies $(\frac{1}{k}, 2-\varepsilon)$ exponential polarization.*

Proof. Take $P \in \mathbb{F}_q^{k\times k}$ and $T \in \mathbb{F}_q^{k\times 4}$ as in the definition of containment. Let moreover j be the last non-zero row of T. We have

$$\begin{aligned}\overline{H}((UM)_j|(UM)_{<j}, A) &= \overline{H}((UM)_j T_{j,4} + (UM)_{<j}T_{<j,4}|(UM)_{<j}, A)\\ &= \overline{H}((UMT)_4|(UM)_{<j}, A)\\ &\le \overline{H}((UMT)_4|(UM)_{<j}T_{<j,<4}, A).\end{aligned}$$

Observe now that $(UM)_{<j}T_{<j,<4} = (UMT)_{<4}$. Indeed — according to the definition of useful containment and because j is last non-zero row of T, we have $T_{j,<4} = 0$ (j-th row has only one non-zero entry $T_{j,4}$, as well as $T_{>j,<4} = 0$. Therefore

$$\begin{aligned}\overline{H}((UM)_j|(UM)_{<j}, A) &\le \overline{H}((UMT)_4|(UMT)_{<4}, A)\\ &= \overline{H}((UP^{-1}R)_4|(UP^{-1}R)_{<4}, A)\\ &= \overline{H}((UR)_4|(UR)_{<4}, A),\end{aligned}$$

where the last equality follows from the fact that U and UP^{-1} are identically distributed (i.e. entries in U are i.i.d).

This conditional entropy was bounded in the proof of Lemma 19. ◀

3.2 Maximally polarizing matrix

In this subsection we will prove Lemma 10.

Proof of Lemma 10. Let us again consider a sequence of i.i.d. pairs (U_i, A_i) for $i \in [k]$, such that $H(U_i|A_i) = \delta$. By Lemma 31, there is some $f : \Sigma \to \mathbb{F}_q$ such that $\Pr(f(A_i) \neq U_i) \le \delta$. Let us take $\tilde{U}_i := U_i + f(A_i)$.

We wish to bound $\overline{H}((UM)_j|(UM)_{<j}, A)$, for all $j > (1-\eta)k$. We have

$$\overline{H}((UM)_j|(UM)_{<j}, A) \le \overline{H}(U|UM_0, A) = H(\tilde{U}|\tilde{U}M_0, A) \le \overline{H}(\tilde{U}|\tilde{U}M_0),$$

where the inequalities follow from the fact that for random variables (X, Y, S, T) it is always the case that $\overline{H}(X|S,T) \le \overline{H}(X,Y|S,T) \le \overline{H}(X,Y|S)$.

Given $\tilde{U}M_0$ we can produce estimate $\hat{U} := \operatorname{argmin}_V\{\mathrm{wt}(V) : VM_0 = \tilde{U}M_0\}$, where $\mathrm{wt}(V) = |\{j : V_j \neq 0\}|$.

Let us observe that if $\mathrm{wt}(\tilde{U}) \le b$ then $\hat{U} = \tilde{U}$. Indeed, we have $\mathrm{wt}(\hat{U}) \le \mathrm{wt}(\tilde{U})$, therefore $\mathrm{wt}(\hat{U} - \tilde{U}) \le 2\mathrm{wt}(\tilde{U}) \le 2b$, but on the other hand $(\hat{U} - \tilde{U})M_0 = 0$, and by the assumption on $\ker M_0^T$ we deduce that $\hat{U} - \tilde{U} = 0$. Therefore $\Pr(\tilde{U} \neq \hat{U}) \le \Pr(\mathrm{wt}(\tilde{U}) > b)$. All coordinates of $\tilde{U}$ are independent, and each $\tilde{U}_i$ is nonzero with probability at most δ, therefore

$$\Pr(\mathrm{wt}(\tilde{U}) > \beta_1) \le \binom{k}{b}\delta^b$$

and by Fano inequality (Lemma 32), we have

$$H(\tilde{U}|\tilde{U}M_0) \le 2C\delta^b(b\log\delta^{-1} + b\log C + \log q)$$

where $C = \binom{k}{b}$. Again, for any ε, and small enough δ (with respect to ε, b, C, q), we have $H(\tilde{U}|\tilde{U}M_0) \le \delta^{\beta_1-\varepsilon}$.

This shows that for any $j > (1-\eta)k$ and small enough δ we have

$$H((UM)_j|(UM)_{<j}, A) \le \delta^{b-\varepsilon},$$

which completes the proof of a exponential polarization for matrix M. ◀

3.3 Source coding implies good distance

Proof of Lemma 13. Consider maximum likelihood decoder $\mathrm{Dec}'(y) := \mathrm{argmax}_{x\in\mathbb{F}_q^k} \Pr(U = x|UP = y)$. By definition, we have $\Pr(\mathrm{Dec}'(UP) \ne U) < \Pr(\mathrm{Dec}(UP) \ne U) < \exp(-k^\gamma)$.

Note that for U distributed according to $B_q(\varepsilon)$, we have $\mathrm{Dec}'(y) = \mathrm{argmin}_{x:xM=y} \mathrm{wt}(x)$, where $\mathrm{wt}(x)$ is number of non-zero elements of x.

Consider set $E = \{x \in \mathbb{F}_q^k : \exists h \in \ker M, \mathrm{wt}(x+h) < \mathrm{wt}(x)\}$, and observe that $\Pr(\mathrm{Dec}'(UM) \ne U) \ge \Pr(U \in E)$. We say that vector $u \in \mathbb{F}_q^k$ is *dominated* by $v \in \mathbb{F}_q^k$ (denoted by $u \preceq v$) if and only if $\forall i \in \mathrm{supp}(u) u_i = v_i$. We wish to argue that for any $w_1 \in E$ and any $w_2 \succeq w_1$, we have $w_2 \in E$. Indeed, if $w_1 \in E$, then there is some $h \in \ker M$ such that $\mathrm{wt}(w_1+h) < \mathrm{wt}(w_1)$. We will show that $\mathrm{wt}(w_2+h) < \mathrm{wt}(w_2)$, which implies that $w_2 \in E$. Given that $w_1 \preceq w_2$, we can equivalently say that there is a vector d with $w_1 + d = w_2$ and $\mathrm{wt}(w_2) = \mathrm{wt}(w_1) + \mathrm{wt}(d)$. Hence

$$\mathrm{wt}(w_2+h) = \mathrm{wt}(w_1+d+h) \le \mathrm{wt}(w_1+h)+\mathrm{wt}(d) < \mathrm{wt}(w_1)+\mathrm{wt}(d) = \mathrm{wt}(w_1+d) = \mathrm{wt}(w_2)$$

Consider now $w_0 \in \ker P$ to be minimum weight non-zero vector, and let us denote $A = \mathrm{wt}(w_0)$. We wish to show a lower bound for A. By definition of the set E we have $w_0 \in E$, and by upward closure of E with respect to domination we have $\Pr(U \in E) \ge \Pr(w_0 \preceq U) = (\frac{\varepsilon}{q-1})^A$.

On the other hand we have $\Pr(U \in E) \le \Pr(\mathrm{Dec}'(UP) \ne U) \le \Pr(\mathrm{Dec}(UP) \ne U) \le \exp(-k^\gamma)$. By comparing these two inequalities we get

$$A \ge \frac{k^\gamma}{\log(q/\varepsilon)} \ .$$ ◀

4 Strong polarization from limiting exponential polarization, generically

Suppose we know that polar codes associated with a matrix $M \in \mathbb{F}_q^{k\times k}$ achieve capacity with error probability $\exp(-N^\beta)$ in the limit of block lengths $N \to \infty$. In this section, we prove a general result that 'lifts" (in a black box manner) such a statement to the claim that, for any $\beta' < \beta$, polar codes associated with M achieve polynomially fast convergence to capacity (i.e., the block length N can be as small as $\mathrm{poly}(1/\varepsilon)$ for rates within ε of capacity), and $\exp(-N^{\beta'})$ decoding error probability *simuletaneously*. Thus convergence to capacity at finite block length comes with almost no price in the failure probability. Put differently, the result states that one can get polynomial convergence to capacity for free once one has a proof of convergence to capacity in the limit with good decoding error probability. This latter fact was shown in [7] for the binary alphabet and [8] for general alphabets.

Proof. Proof of 18 Consider the channel that outputs $X + Z$ on input X, where $Z \sim B_q(\gamma)$ for some $\gamma > 0$ (depending on β, β'). The hypothesis on M implies that for sufficiently large N the polar code corresponding to M will have failure probability at most $\exp(-N^\beta)$ on this channel. Using the well-known equivalence between correcting errors for this additive channel, and linear compression schemes, we obtain that for all large enough t there is some subset S of $(h_q(\gamma) + \varepsilon)k^t$ columns of $M^{\otimes t}$ that defines a linear compression scheme (for k^t i.i.d copies of $B_q(\gamma)$), along with an accompanying decompression scheme with error probability (over the randomness of the source) at most $\exp(-k^{\beta t})$.

We now claim that for all $\beta' < \beta$, there exists $t_0 = t_0(\beta', \beta)$ such that the Arikan martingale associated with some column permuted version of $M^{\otimes t_0}$, is $\beta' t_0 \log_2 k$-exponentially strongly polarizing.

The proof of this claim is in fact immediate, given the ingredients developed in previous sections. Apply the hypothesis about M in the theorem with the choice $\varepsilon = (\beta - \beta')/4$ and γ chosen small enough as a function β, β' so that $h_q(\gamma) \leq (\beta - \beta')/4$ and let t_0 be a large enough promised value of t. Put $m = k^{t_0}$, and $\ell = (h_q(\gamma) + \varepsilon)m$ and $L = M^{\otimes t_0}$. Using Lemma 13, we know there is submatrix $L' \in \mathbb{F}_q^{m \times \ell}$ of L such that $\ker((L')^T)$ defines a code of distance $\Delta \geq m^\beta / \log^{-1}(q/\gamma)$. Define $M_0 = [L' \mid \cdot] \in \mathbb{F}_q^{m \times m}$ to be any matrix obtained by permuting the columns of L such that the columns in L' occur first. By Lemma 10, the matrix M_0 is $(1 - \ell/m, \Delta)$-polarizing. For our choice of γ, ε, $\ell/m \leq \frac{\beta - \beta'}{2}$ and $\Delta \geq m^{(\beta+\beta')/2}$. using Lemma 7 and Theorem 5, it follows that the Arikan martingale associated with M_0 exhibits $(\beta+\beta')/2 \times \left(1 - \frac{\beta-\beta'}{2}\right) \log_2 m$-exponentially strong polaraization. Since $(\beta + \beta')/2 \times \left(1 - \frac{\beta-\beta'}{2}\right) \geq \beta'$, the claim follows.

Applying Theorem 15 to the matrix $M_0 = M^{\otimes t_0}$ we conclude that there is a polynomial p such that given the gap to capacity $\varepsilon > 0$, and for every s satisfying $N = k^{t_0 s} \geq \text{poly}(\frac{1}{\varepsilon})$ there is an affine code generated by a subset of rows of $(M_0^{-1})^{\otimes s}$ which achieves ε-gap to capacity and has failure probability $\exp(-N^{\beta'})$. But this resulting code is simply an affine code generated by a subset of the rows of $(M^{-1})^{\otimes t}$, for $t = st_0$, This concludes the proof. ◀

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 Jaroslaw Blasiok, Venkatesan Guruswami, Preetum Nakkiran, Atri Rudra, and Madhu Sudan. General strong polarization. *CoRR*, abs/1802.02718, 2018. `arXiv:1802.02718`.

3 Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. John Wiley and Sons, Hoboken, NJ, USA, 2nd edition, 2005.

4 Venkatesan Guruswami and Ameya Velingker. An entropy sumset inequality and polynomially fast convergence to Shannon capacity over all alphabets. In *Proceedings of 30th Conference on Computational Complexity*, pages 42–57, 2015.

5 Venkatesan Guruswami and Patrick Xia. Polar codes: Speed of polarization and polynomial gap to capacity. In *Proceedings of the 54th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 310–319, October 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 Ryuhei Mori and Toshiyuki Tanaka. Source and channel polarization over finite fields and reed-solomon matrices. *IEEE Trans. Information Theory*, 60(5):2720–2736, 2014.

A Local to global exponential polarization

The proof of Theorem 5 is essentially the same as the proof of corresponding Theorem 1.6 in [2]. Lemma 26 and Lemma 27 are new in this paper, yet the proof of Lemma 26 is similar to the proof of Lemma 3.3 there. Theorem 5 is essentially repeating the argument from Theorem 1.6 in [2], except for using Lemma 27 in place of the lemma present therein, and hence arriving at stronger conclusion.

We remind a definition of adapted sequence from [2].

▶ **Definition 25.** We say that a sequence $Y_1, Y_2 \dots$ of random variables is *adapted* to the sequence $X_1, X_2 \dots$ if and only if for every t, Y_t is completely determined given $X_1, \dots X_t$. We will use $\mathbb{E}[Z|X_{[1:t]}]$ as a shorthand for $\mathbb{E}[Z|X_1, \dots X_t]$, and $\Pr[E|X_{[1:t]}]$ as a shorthand for $\mathbb{E}[\mathbf{1}_E|X_1, \dots X_t]$. If the underlying sequence X is clear from context, we will skip it and write just $\mathbb{E}[Z|\mathcal{F}_t]$.

▶ **Lemma 26.** *There exist $C < \infty$ such that for all η, b, ε following holds. Let X_t be a martingale satisfying $\Pr(X_{t+1} < X_t^b | X_t) \geq \eta$, where $X_0 \in (0,1)$. Then*

$$\Pr(\log X_T > (\log X_0 + CT) b^{(1-\varepsilon)\eta T}) < \exp(-\Omega(\varepsilon\eta T))$$

Proof. Let us consider random variables $Y_t := \log(X_t/X_{t-1})$. This sequence of random variables is adapted to the sequence X_t in the sense of Definition 25. Let us decompose $Y_t = Y_t^+ + Y_t^-$, where $Y_t^+ = Y_t \mathbf{1}_{Y_t \geq 0}$. Note that by Markov inequality

$$\Pr(Y_{t+1} > \lambda | X_{[1:t]}) = \Pr(X_{t+1} > X_t \exp(\lambda) | X_{[1:t]}) \leq \exp(-\lambda) \frac{\mathbb{E}[X_{t+1}|X_{[1:t]}]}{X_t} = \exp(-\lambda)$$

By Lemma 34 we deduce that for some C, we have

$$\Pr(\sum_{i \leq T} Y_i^+ > CT) \leq \exp(-\Omega(T))$$

On the other hand, if we take Z_t to be the indicator variable for an event $X_t < X_{t-1}^{\beta_1}$. By Lemma 35 we have

$$\Pr(\sum_{i \leq T} Z_i \leq (1-\varepsilon)\eta T) \leq \exp(-\Omega(T\varepsilon\eta))$$

If both of those unlikely events do not hold, that is we have simultaneously $\sum_{i \leq T} Y_i^+ < CT$ and $\sum_{i \leq T} Z_i > (1-\varepsilon)\eta T$, we can deduce that $\log X_T \leq (\log X_0 + CT) b^{(1-\varepsilon)\eta T}$ — i.e. the largest possible value of X_T is obtained if all the initial Y_i were positive and added up to CT (at which point value of the martingale would satisfy $\log X_{T'} \leq \log X_0 + CT$), followed by $(1-\varepsilon)\eta T$ steps indicated by variables Z_i — for each of those steps, $\log X_{t+1} \leq b \log X_t$. ◀

▶ **Lemma 27.** *For all $\eta, b, \varepsilon, \gamma$ the following holds. Let X_t be a martingale satisfying $\Pr(X_{t+1} < X_t^b | X_t) \geq \eta$, where $X_0 < \exp(-\gamma T)$ with some $\gamma > 0$, then*

$$\Pr(\log X_T < -b^{(1-\varepsilon)\eta T}) < \exp(-\Omega_{\nu,\varepsilon,\eta,\gamma}(T)))$$

Proof. Consider sequence $t_0, t_1, \dots t_m \in [T]$, where $t_0 = 0, t_m = T$, and $\frac{\gamma T}{C} \leq |t_i - t_{i-1}| \leq \frac{\gamma T}{2C}$, and therefore $m = \mathcal{O}(C\gamma^{-1})$, where C is a constant appearing in the statement of Lemma 26. For each index $s \in [m]$ we should consider a martingale $X_i^{(s)} := X_{t_s+i}$, and we wish to apply Lemma 26 to this martingale $\hat{X}^{(s)}$, with $T = t_{s+1} - t_s$. We can union bound total failure probability by $m \exp(-\Omega(\gamma\varepsilon\eta T))$.

In case we succeed, we can deduce that for each i we have

$$\log X_{t_i} < (\log X_{t_{i-1}} + C(t_i - t_{i-1}))b^{(1-\varepsilon)\eta(t_i - t_{i-1})}. \tag{1}$$

We will show that by our choice of parameters, we can bound $C(t_i - t_{i-1}) \leq -\frac{1}{2}\log X_{t_i}$. Let us first discuss how this is enough to complete the proof. Indeed, in such a case we have

$$\log X_{t_i} < \frac{1}{2}(\log X_{t_{i-1}})b^{(1-\varepsilon)\eta(t_i - t_{i-1})}, \tag{2}$$

and by induction

$$\log X_{t_m} < \frac{1}{2^m}(\log X_0)b^{(1-\varepsilon)\eta t_m}.$$

For fixed η, m and T large enough (depending on η, m, ε), this yields $\log X_T < -b^{(1-2\varepsilon)\eta T}$, and the result follows up by changing ε by a factor of 2.

All we need to do is to show is that for every i we have

$$C(t_{i+1} - t_i) \leq -\frac{1}{2}\log X_{t_i}, \tag{3}$$

assuming that inequalities (1) hold for every i. We will show this inductively, together with $\log X_{t_i} \leq -\gamma T$. Note that we assumed this inequality to be true for $X_{t_0} = X_0$. By our choice of parameters we have $C(t_{i+1} - t_i) \leq \frac{\gamma T}{2}$, therefore for t_{i+1} the inequality (3) is satisfied.

We will now show that $\log X_{t_{i+1}} \leq \log X_{t_i} \leq -\gamma T$ to finish the proof by induction. We can apply inequality (2) to X_{t_i}, to deduce that $\log X_{t_{i+1}} \leq \frac{1}{2}(\log X_{t_i})b^{\frac{1}{2}\frac{\gamma}{C}T}$. This for large values of T (given parameters b, γ and C) yields $\log X_{t_{i+1}} < \log X_{t_i}$ — indeed this inequality will be true as soon as $b^{\frac{\gamma}{2C}T} > 2$, because both $\log X_{t_{i+1}}$ and $\log X_{t_i}$ are negative, which completes the proof. ◀

Before we proceed with the proof, let us recall the following lemma from [2], stating that locally polarizing martingales are exponentially close to boundary $\{0, 1\}$ for some basis $(1 - \nu)$, except with exponentially small failure probability.

▶ **Lemma 28** (Lemma 3.1 from [2]). *If a $[0, 1]$-martingale sequence $X_0, \ldots X_t, \ldots$, satisfies $(\alpha, \tau(\cdot), \theta(\cdot))$-local polarization, then there exist $\nu > 0$, depending only on α, τ, θ, such that*

$$\mathbb{E}[\min(\sqrt{X_t}, \sqrt{1 - X_t})] \leq (1 - \nu)^t.$$

We will also need Lemma 3.3 from [2] — it plays the same role as Lemma 27 to control strong polarization of the martingale at the high end (where the exponential suction condition does not apply).

▶ **Lemma 29** (Lemma 3.3 from [2]). *There exists $c < \infty$, such that for all K, α with $K\alpha \geq c$ the following holds. Let X_t be a martingale satisfying $\Pr\left(X_{t+1} < e^{-K}X_t | X_t\right) \geq \alpha$, where $X_0 \in (0, 1)$. Then $\Pr(X_T > \exp(-\alpha KT/4)) \leq \exp(-\Omega(\alpha T))$.*

We are now ready to prove local to global lifting theorem for exponential polarization.

Proof of Theorem 5. Consider locally polarizing martingale, and let us fix some $\varepsilon > 0$. By Markov inequality applied to 28 with $t = \varepsilon T$ we deduce that for some ν we have

$$\Pr(\max(X_{\varepsilon T}, 1 - X_{\varepsilon T}) \geq (1 - \frac{\nu}{4})^{\varepsilon T}) < \exp(-\Omega_\varepsilon(T))$$

Consider τ_0 to be such that if $X_t < \tau_0$, we have probability at most η that $X_{t+1} < X_t^b$ (existence of such a value is guaranteed by exponential local polarization), and moreover if $1 - X_t < \tau_0$, we have probability at least α for $(1 - X_{t+1}) < \exp(-K)(1 - X_{t+1})$, where K is large constant depending on α and the target rate of polarization — this is guarantee by suction at the high end condition in local polarization definition of a martingale X_t.

Let us condition on $\max(X_{\varepsilon T}, 1 - X_{\varepsilon T}) < (1 - \frac{\nu}{4})^{\varepsilon T}$. By the Doobs martingale inequality (Lemma 33), we can deduce that $\Pr(\max_{t\in[\varepsilon T,T]} \max(X_t, 1 - X_t) > \tau) \leq \tau^{-1}(1 - \frac{\nu}{4})^{-\varepsilon T} \leq \exp(-\Omega_{\tau,\nu,\varepsilon}(T))$. Let us now condition in turn on this event not happening.

We will consider first the case when $X_{\varepsilon T} < (1 - \frac{\nu}{4})^{\varepsilon T}$, and let us put $\gamma := -\varepsilon \log(1 - \frac{\nu}{4})$, so that $X_{\varepsilon T} < \exp(-\gamma T)$.

We can now apply Lemma 27 to the martingale sequence starting with $X_{\varepsilon T}$ — the assumption of those lemmas are satisfied, as long as X_t stays bounded by τ (by the exponential local polarization property), hence we deduce that in this case, except with probability $\exp(-\Omega_{\gamma,\varepsilon,\eta}(T))$, we have

$$\log X_T < -b^{(1-\varepsilon)^2\eta T},$$

and therefore $X_T < 2^{-b^{(1-\varepsilon)^2\eta T}}$.

On the other hand, if $1 - X_t < \tau$ for all $\varepsilon T \leq t \leq T$, the suction at the high end condition of local polarization applies, and we can apply Lemma 29 to martingale $1 - X_{\varepsilon T+t}$ to deduce that except with probability $\exp(-\Omega_\alpha(T))$, we have $1 - X_T < \exp(-\alpha K(1 - \varepsilon T)/4) < \gamma^T$ for suitable choice of K depending on γ and α. ◀

B Arikan Martingale

In this section, we provide a definition of Arikan Martingale.

For every matrix invertible matrix M and channel $\mathcal{C} : \mathbb{F}_q \to \mathcal{Y}$, we define a martingale sequence X_t, for $t = 0, 1, \ldots$, where all $X_t \in [0, 1]$.

Intuitively, for a given matrix M and $t \in \mathbb{N}$, the marginal distribution of X_t is the same as distribution of $\overline{H}((\boldsymbol{Z}M^{\otimes t})_j \,|\, (\boldsymbol{Z}M^{\otimes t})_{<j}, \boldsymbol{Y})$ over a random index $j \in [k^t]$, where $\boldsymbol{Z}_i \sim \mathrm{Unif}(\mathbb{F}_q)$ are independent, and $\mathcal{Y}_i$ sampled independently according to $Y_i \sim C_{Y|Z=\boldsymbol{Z}_i}$. That is, we apply matrix $M^{\otimes t}$ to a vector with independent coordinates Z_i, and we look at the entropy of the random output coordinate, conditioned on all previous ones. The entries Z_i, conditioned on Y_i have normalized entropy equal to $1 - \mathrm{Capacity}(C)$ for symmetric channel C, in particular $X_0 = 1 - \mathrm{Capacity}(C)$. If the variable X_t is strongly polarized, it means that about $1 - \mathrm{Capacity}(C)$ fraction of all $(\boldsymbol{Z}M^{\otimes t})_j$ have entropy close to one (after conditioning on all the previous entries), and most of remaining variables has entropy close to zero — they can be predicted from the previous values with huge probability.

The martingale structure of X_t with respect to t is a consequence of chain rule for entropy, together with recursive decomposition of multiplication by matrix $M^{\otimes t}$. The relation between our definition of exponential matrix polarization (Definition 6) and the local behavior of the Arıkan martingale is consequence of the fact that $\boldsymbol{A'}$ (in the definition below) is obtained from independent copies $\boldsymbol{A}$ via multiplication by $\boldsymbol{M}$. The notational difficulty in proving this equivalence (Lemma 7) follows from the fact that conditioning in the conditional entropies under consideration is syntatically different — although equivalent.

In what follows, the vectors in $\mathbb{F}_q^{k^t}$ are indexed by tuples $\boldsymbol{j} \in [k]^t$, $\preceq$ denotes a lexicographic order on tuples. For $\boldsymbol{A} \in \mathbb{F}_q^{k^t}$ and $\boldsymbol{j} \in [k]^t$, we use notation $\boldsymbol{A}_{\preceq \boldsymbol{j}}$ to denote all entries of A with indices preceeding j according to lexicographic order $\preceq$. Moreover for a tuple of

indices $\boldsymbol{j} \in [k]^{t-1}$, and a vector $\boldsymbol{A} \in \mathbb{F}_q^{k^t}$, we use notation $A_{[\boldsymbol{j},\cdot]} \in \mathbb{F}_q^k$ to denote a vector $(A_{[\boldsymbol{j},1]}, \dots A_{[\boldsymbol{j},k]})$.

▶ **Definition 30** (Arıkan martingale, Defintion 4.1 in [2]). Given an invertible matrix $M \in \mathbb{F}_q^{k\times k}$ and a channel description $C_{Y|Z}$ for $Z \in \mathbb{F}_q, Y \in \mathcal{Y}$, the Arıkan-martingale $X_0, \dots X_t, \dots$ associated with it is defined as follows. For every $t \in \mathbb{N}$, zlet D_t be the distribution on pairs $\mathbb{F}_q^{k^t} \times \mathcal{Y}^{k^t}$ described inductively below:

A sample (A, B) from D_0 supported on $\mathbb{F}_q \times \mathcal{Y}$ is obtained by sampling $A \sim \mathbb{F}_q$, and $B \sim C_{Y|Z=A}$. For $t \geq 1$, a sample $(\boldsymbol{A'}, \boldsymbol{B'}) \sim D_t$ supported on $\mathbb{F}_q^{k^t} \times \mathcal{Y}^{k^t}$ is obtained as follows:

- Draw k independent samples $(\boldsymbol{A}^{(1)}, \boldsymbol{B}^{(1)}), \dots, (\boldsymbol{A}^{(k)}, \boldsymbol{B}^{(k)}) \sim D_{t-1}$.
- Let $\boldsymbol{A'}$ be given by $\boldsymbol{A'}_{[\boldsymbol{i},\cdot]} = (\boldsymbol{A}_{\boldsymbol{i}}^{(1)}, \dots, \boldsymbol{A}_{\boldsymbol{i}}^{(k)}) \cdot M$ for all $\boldsymbol{i} \in [k]^{t-1}$.
- Take $\boldsymbol{B'} = (\boldsymbol{B}^{(1)}, \boldsymbol{B}^{(2)}, \dots \boldsymbol{B}^{(k)})$.

Then, the sequence X_t is defined as follows: For each $t \in \mathbb{N}$, sample $i_t \in [k]$ iid uniformly. Let $\boldsymbol{j} = (i_1, \dots, i_t)$ and let $X_t := \overline{H}(\boldsymbol{A}_{\boldsymbol{j}} | \boldsymbol{A}_{\prec \boldsymbol{j}}, \boldsymbol{B})$, where the entropies are with respect to the distribution $(\boldsymbol{A}, \boldsymbol{B}) \sim D_t$. The only randomness in the process X_t comes from the selection of random multi-index $\boldsymbol{j}$.

C Standard probabilistic inequalities

▶ **Lemma 31** (Lemma 2.2 in [2]). *For a pair of random variables $(U_1, U_2) \in \Sigma_1 \times \Sigma_2$ there exists function $f : \Sigma_2 \to \Sigma_1$ such that $\Pr(f(U_2) \neq U_1) \leq H(U_1|U_2)$.*

▶ **Lemma 32** (Fano's inequality). *For a pair of random variables $(U_1, U_2) \in \Sigma_1 \times \Sigma_2$, if we have a function $f : \Sigma_2 \to \Sigma_1$ such that $\Pr(f(U_2) \neq U_1) \leq \delta$ with $\delta < \frac{1}{2}$, then $H(U_2|U_1) \leq 2\delta(\log \delta^{-1} + \log \Sigma_1)$.*

▶ **Lemma 33** (Doobs martingale inequality). *For any non-negative martingale X, we have*

$$\Pr(\sup_{t \leq T} X_t > \lambda) \leq \lambda^{-1} X_0$$

We include the statements of following lemmas from [2] for reference.

▶ **Lemma 34.** *Consider a sequence of non-negative random variables $Y_1, Y_2, \dots, Y_t, \dots$ adapted to the sequence X_t. If for every t we have $\Pr(Y_{t+1} > \lambda \,|\, X_{[1:t]}) < \exp(-\lambda)$, then for every $T > 0$:*

$$\Pr(\sum_{i \leq T} Y_i > CT) \leq \exp(-\Omega(T))$$

for some universal constant C.

▶ **Lemma 35.** *Consider a sequence of random variables $Y_1, Y_2, \dots$ with $Y_i \in \{0, 1\}$, adapted to the sequence X_t. If $\Pr(Y_{t+1} = 1 | X_{[1:t]}) > \mu_{t+1}$ for some deterministic value μ_t, then for $\mu := \sum_{t \leq T} \mu_t$ we have*

$$\Pr(\sum_{t \leq T} Y_t < (1 - \varepsilon)\mu) \leq \exp(-\Omega(\varepsilon\mu))$$

Approximate Degree and the Complexity of Depth Three Circuits

Mark Bun[1]
Princeton University, Princeton, NJ, USA
mbun@cs.princeton.edu

Justin Thaler[2]
Georgetown University, Washington, DC, USA
justin.thaler@georgetown.edu

Abstract

Threshold weight, margin complexity, and Majority-of-Threshold circuit size are basic complexity measures of Boolean functions that arise in learning theory, communication complexity, and circuit complexity. Each of these measures might exhibit a *chasm* at depth three: namely, all polynomial size Boolean circuits of depth two have polynomial complexity under the measure, but there may exist Boolean circuits of depth three that have essentially maximal complexity $\exp(\Theta(n))$. However, existing techniques are far from showing this: for all three measures, the best lower bound for depth three circuits is $\exp(\tilde{\Omega}(n^{2/5}))$. Moreover, prior methods exclusively study *block-composed* functions. Such methods appear intrinsically unable to prove lower bounds better than $\exp(\Omega(\sqrt{n}))$ even for depth four circuits, and have yet to prove lower bounds better than $\exp(\tilde{\Omega}(\sqrt{n}))$ for circuits of any constant depth.

We take a step toward showing that all of these complexity measures indeed exhibit a chasm at depth three. Specifically, for any arbitrarily small constant $\delta > 0$, we exhibit a depth three circuit of polynomial size (in fact, an $O(\log n)$-decision list) of complexity $\exp(\Omega(n^{1/2-\delta}))$ under each of these measures.

Our methods go beyond the block-composed functions studied in prior work, and hence may not be subject to the same barriers. Accordingly, we suggest natural candidate functions that may exhibit stronger bounds.

2012 ACM Subject Classification Theory of computation

Keywords and phrases approximate degree, communication complexity, learning theory, polynomial approximation, threshold circuits

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.35

Related Version The full version of this work is available at `https://eccc.weizmann.ac.il/report/2016/121/`.

1 Introduction

Let $f\colon \{-1,1\}^n \to \{-1,1\}$ be a Boolean function, and let $\mathcal{C}\colon 2^{\{-1,1\}^n} \to \mathbb{N}$ denote a measure of the complexity of f. We say that $\mathcal{C}$ exhibits a *chasm* at depth three if all Boolean circuits[3] of depth two have polynomial complexity under the measure, but there exist circuits of depth

[1] This work was done while the author was at Harvard University and visiting Yale University, supported by an NDSEG Fellowship and NSF grant CNS-1237235.
[2] Parts of this work were performed while the author was at Yahoo Research.
[3] Throughout this paper, unless otherwise noted, all circuits under consideration are assumed to have polynomial size, and to be over the basis AND, OR and NOT.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018).
Editors: Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer; Article No. 35; pp. 35:1–35:18

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

three that have essentially maximal complexity $\exp(\Theta(n))$. Examples of measures that may satisfy a chasm at depth three include:

- **Threshold Weight.** A polynomial $p\colon \{-1,1\}^n \to \mathbb{R}$ with integer coefficients is said to sign-represent f if $p(x) \cdot f(x) > 0$ for all $x \in \{-1,1\}^n$. The weight of p, denoted $W(p)$, is the sum of the absolute value of its coefficients. The *threshold weight* of f is the least weight of a sign-representing polynomial for f.
  It is easy to see that all DNF and CNF formulae of size s have threshold weight $O(ns)$. The best known upper bound on the threshold weight of depth three circuits is the trivial $2^{O(n)}$ bound. Hence, threshold weight may exhibit a chasm at depth three.
- **Discrepancy and Margin Complexity.** Discrepancy, defined formally in Appendix A.4, is a central quantity in communication complexity and circuit complexity.[4] For example, discrepancy is known to characterize the communication complexity class **PP** [13], and small discrepancy implies large communication complexity in nearly every communication model. The multiplicative inverse of discrepancy is also known to be equivalent to *margin complexity*, a central quantity in learning theory [20].
  All DNF and CNF formulae have at least inverse-polynomial discrepancy. However, the best known lower bound on the discrepancy of depth three circuits is the trivial $2^{-O(n)}$ bound. Hence, margin complexity and (the inverse of) discrepancy may exhibit a chasm at depth three.
- **Majority-of-Threshold Circuit Size**. Since OR and AND can each be computed by a single Majority gate, all DNF and CNF formulae are computed by Majority-of-Threshold (in fact, Majority-of-Majority) circuits of polynomial size. Meanwhile, the best known upper bound on the size of Majority-of-Threshold circuits computing depth three Boolean circuits is the trivial $2^{O(n)}$ bound. Hence, Majority-of-Threshold circuit size may exhibit a chasm at depth three.

We discuss each of these measures, together with applications in communication complexity and learning theory, in more detail in Appendix A.

Unfortunately, we are currently quite far from proving that any of the above complexity measures actually exhibit such a chasm. For each measure, the best known lower bound for depth three circuits is $\exp(\tilde{\Omega}(n^{2/5}))$ [8]. Moreover, as we explain in Section 1.2.4, existing techniques appear intrinsically unable to prove a lower bound better than $\exp(\Omega(n^{1/2}))$ even for circuits of depth four. This barrier stems from the fact that previous work has focused exclusively on analyzing *block-composed* functions. Here, a function $f\colon \{-1,1\}^{N \cdot M} \to \{-1,1\}$ is said to be block-composed if there are two functions $h\colon \{-1,1\}^N \to \{-1,1\}$ and $g\colon \{-1,1\}^M \to \{-1,1\}$ such that $f = h \circ g := h(g,\ldots,g)$. That is, a function is block-composed if it interprets its input as a sequence of N *blocks* $x_1,\ldots,x_N \in \{-1,1\}^M$, applies a Boolean function g independently to each block x_i, and then feeds the N outputs into a different function h.

In this paper, we take a step toward showing that all three of these complexity measures indeed exhibit a chasm at depth three. Specifically, for any constant $\delta > 0$, we exhibit a depth three circuit of polynomial size (in fact, an $O(\log n)$-decision list) of complexity $\exp(\Omega(n^{1/2-\delta}))$ under each of these measures. Our improvement over prior work stems from the fact that we move beyond block-composed functions, and hence our methods may not be subject to the same barriers. In particular, we suggest natural candidate functions that

[4] Discrepancy is often thought of as a matrix-analytic quantity, rather than as a Boolean function complexity measure. For a function $f\colon \{-1,1\}^n \times \{-1,1\}^n \to \{-1,1\}$, when we refer to the discrepancy of f, we mean the discrepancy of the matrix $[f(x,y)]_{x,y \in \{-1,1\}^n}$.

may exhibit stronger bounds, of the form $\exp(\tilde{\Omega}(n))$, where the $\tilde{\Omega}$ notation hides factors polylogarithmic in n (cf. Section 3.3).

The functions that underly our analysis are rather complicated to define, but here we briefly highlight their novel features. Inspired by prior work of Podolskii [24] (see Section 1.2.2 for further discussion), we define our functions to be "almost" block-composed, but to have mild dependencies between blocks. Just like a block-composed function, each of our functions interprets its input as a sequence of N blocks $x_1, \ldots, x_N \in \{-1,1\}^M$, and applies a Boolean function g to each block, before feeding the N outputs into a different function h. However, for $i \geq 2$, before applying g to x_i we first pass x_i through a "pre-processing function" that *depends on the preceding blocks* $x_1, x_2, \ldots, x_{i-1}$. This dependency is simple enough that the final function is computed by a circuit of depth three, but complicated enough that the function has much higher complexity than the block-composed functions considered in prior work.

1.1 Our Contributions: Details

The three complexity measures $\mathcal{C}$ described above are intimately related to uniform approximability by low-degree polynomials, as we now explain. Roughly speaking, for each of the three measures, in order to construct a function of complexity at least 2^d under $\mathcal{C}$, it suffices to identify a function $f\colon \{-1,1\}^n \to \{-1,1\}$ such that f cannot be uniformly approximated to error $1-2^{-d}$ by polynomials of degree at most d. One can then apply known transformations [17, 30, 16] to transform f into a related function $F\colon \{-1,1\}^n \to \{-1,1\}$ such that $\mathcal{C}(F) \geq 2^{\Omega(d)}$. Moreover, these transformations are simple in the following sense: if f is computed by a (polynomial size) depth d circuit with logarithmic bottom fan-in, so is F.

Accordingly, the main technical contribution of this paper is to prove a new lower bound on the approximability of suitable constant-depth circuits by low-degree polynomials.

▶ **Theorem 1.** *For any arbitrarily small constant $\delta > 0$ and any arbitrarily large constant $\Gamma > 1$, there is an (explicitly given) function $f\colon \{-1,1\}^n \to \{-1,1\}$ that is computed by Boolean circuit of depth three, with logarithmic bottom fan-in, that satisfies the following property. For any polynomial $p\colon \{-1,1\}^n \to \mathbb{R}$ of total degree at most $n^{1/2-\delta}$, there exists some $x \in \{-1,1\}^n$ such that $|p(x) - f(x)| > 1 - 2^{-n^\Gamma}$.*

In fact, the function f in Theorem 1 is much simpler than an arbitrary depth three circuit with logarithmic bottom fan-in; it is an $O(\log n)$-decision list of polynomial length. An $O(\log n)$-decision list is a function whose output is determined by a very simple sequential decision process (essentially, a chain of "if-then-else" statements, where each "if" statement is a conjunction on $O(\log n)$ variables—we give a precise definition in Section 2). Decision lists have been studied intensely in learning theory and complexity theory (see, e.g., [26, 15, 3, 35, 27, 16, 5, 11]).

By combining Theorem 1 with known transformations [17, 30], we obtain a depth three circuit F with very large complexity under the three measures described above. In fact, F itself is computed by an $O(\log n)$-decision list.

▶ **Corollary 2.** *For any constant $\delta > 0$, there is an (explicitly given) function $F\colon \{-1,1\}^n \to \{-1,1\}$ computed by an $O(\log n)$-decision list of polynomial length (and hence also a depth-3 circuit with logarithmic bottom fan-in) such that:*

- *The threshold weight of F is $\exp(\Omega(n^{1/2-\delta}))$.*
- *The discrepancy of F is $\exp(-\Omega(n^{1/2-\delta}))$.*
- *Any Majority-of-Threshold circuit computing F has size $\exp(\Omega(n^{1/2-\delta}))$.*

Table 1 succinctly compares our results to prior work.

Table 1 Comparison of our new bounds for AC^0 to prior work. The circuit depth column lists the depth of the Boolean circuit used to exhibit the bound, and δ denotes an arbitrarily small positive constant. All Boolean circuits are polynomial size.

Reference	Threshold Weight Bound	Discrepancy Bound	Majority-of-Threshold Circuit Size Bound	Circuit Depth
[17]	$\exp(\Omega(n^{1/3}))$	N/A	N/A	3
[29]	N/A	$\exp(-\Omega(n^{1/5}))$	$\exp(\Omega(n^{1/5}))$	3
[5, 30]	N/A	$\exp(-\Omega(n^{1/3}))$	$\exp(\Omega(n^{1/3}))$	3
[8]	$\exp\left(\Omega(n^{2/5})\right)$	$\exp\left(-\Omega(n^{2/5})\right)$	$\exp\left(\Omega(n^{2/5})\right)$	3
[33]	$\exp\left(\Omega(n^{\frac{k-1}{2k-1}})\right)$	$\exp\left(-\Omega(n^{\frac{k-1}{2k-1}})\right)$	$\exp\left(\Omega(n^{\frac{k-1}{2k-1}})\right)$	$k+1$ (for $k \geq 2$)
[28]	$\exp\left(\Omega(n^{1/2})\right)$	$\exp\left(-\Omega(n^{1/2})\right)$	$\exp\left(\Omega(n^{1/2})\right)$	4
This work	$\exp\left(\Omega(n^{1/2-\delta})\right)$	$\exp\left(-\Omega(n^{1/2-\delta})\right)$	$\exp\left(\Omega(n^{1/2-\delta})\right)$	3

1.2 Prior Work

In order to discuss prior work, it is helpful to introduce the notions of approximate degree and threshold degree, which both capture the difficulty of pointwise approximation by low-degree polynomials. The ε-approximate degree of a function f, denoted $\widetilde{\deg}_\varepsilon(f)$, is the least degree of a real polynomial that pointwise approximates f to error ε. By convention, $\widetilde{\deg}_{1/3}(f)$ is denoted simply as $\widetilde{\deg}(f)$ and referred to without qualification as the approximate degree of f (the constant $1/3$ is chosen for aesthetic reasons, and could be replaced with any other constant in $(0,1)$ without affecting the theory in any way). The threshold degree of f, denoted $\deg_\pm(f)$, is the least degree of a real polynomial that sign-represents f at all points (such a polynomial is called a polynomial threshold function (PTF) for f). When appropriate, we also use subscripts to indicate the number of variables over which the function is defined. For example, OR_M denotes the OR function on M inputs.

1.2.1 Early Work on Approximating AC^0 Functions by Polynomials

Minsky and Papert [22] famously proved an $\Omega(n^{1/3})$ lower bound on the threshold degree of the DNF formula $\mathrm{OR}_{n^{1/3}} \circ \mathrm{AND}_{n^{2/3}}$, now known as the Minsky-Papert DNF. Klivans and Servedio [14] proved an essentially matching upper bound of $\tilde{O}(n^{1/3})$ on the threshold degree of *any* polynomial size DNF.

Beigel identified a DNF (in fact, a 1-decision list) known as OMB (short for ODD-MAX-BIT) that has threshold degree 1, but requires large degree to approximate to error bounded away from 1 [3]. OMB will play a central role in this paper, and we define it formally in Section 2. Quantitatively, Beigel showed[5] that for any $d > 0$, there is an $\varepsilon = 1 - 2^{-\Omega(n/d^2)}$ such that $\widetilde{\deg}_\varepsilon(\mathrm{OMB}_n) \geq d$. For any $\varepsilon > 0$, Klivans and Servedio [15] gave an optimal ε-approximating polynomial for any 1-decision list, showing that Beigel's lower bound is asymptotically tight for all $d > 0$.[6]

[5] Beigel describes his result as a lower bound on the *degree-d threshold weight* of OMB_n, which refers to the least weight of a sign-representing polynomial p for f satisfying $\deg(p) \leq d$. However, his argument is easily seen to establish the claimed approximate degree lower bound.

[6] Like Beigel, Klivans and Servedio state their results in terms of degree-d threshold weight. However, their construction is easily seen to imply the claimed upper bound on the approximate degree of OMB_n.

1.2.2 Prior Work of Podolskii

Podolskii pioneered a line of work devoted to proving approximate degree lower bounds that hold even when the error parameter ε is allowed to be *super-exponentially* close to 1 [24, 25].[7] In [25], he showed that for any constant $d \geq 2$, there exists a function of threshold degree d that cannot be uniformly approximated to error ε by polynomials of degree at most d, unless $\varepsilon = 1 - n^{-\Omega(n^d)}$. This result is tight, matching an upper bound proved by Buhrman et al. [5].

Our construction and analysis are inspired by another related result of Podolskii [24]. For any constant $d > 0$, Podolskii identified a function f of threshold degree d such that, even for $D \gg d$, the following holds: f cannot be uniformly approximated by degree D polynomials to error ε, unless ε is superexponentially close to 1. Quantitatively, he showed that for any constant $d > 0$, there exists a DNF (in fact, a d-decision list) f with threshold degree d, yet for any $D < O(n^{1/5}/\log n)$, there exists an $\varepsilon \in 1 - \exp\left(\Omega((n/D^4)^d)\right)$ for which $\widetilde{\deg}_\varepsilon(f) \geq D$. Unfortunately, Podolskii's construction does not yield any new bounds on the complexity measures we are interested in. By introducing new ideas, we are able to prove such improved bounds for depth three circuits.

1.2.3 Translating Approximate Degree Bounds to Complexity Bounds

Several works have focused on transforming approximate degree and threshold degree lower bounds into bounds on the complexity measures that we focus on in this paper (threshold weight, discrepancy/margin complexity, and Majority-of-Threshold circuit size). Krause and Pudlák [17] showed how to take a function f of threshold degree at least d, and turn f into a related function F of threshold weight[8] at least 2^d. By applying this transformation to the Minsky-Papert DNF, Krause and Pudlák obtained a depth three circuit (with constant bottom fan-in) with threshold weight $\exp(\Omega(n^{1/3}))$.

Subsequent work by Krause [16] showed that for F to have threshold weight $2^{\Omega(d)}$, it is enough for f to satisfy $\widetilde{\deg}_{1-2^{-d}}(f) \geq d$.[9] Krause applied his result to the function $f = \mathrm{OMB}$, to obtain an $\exp(\Omega(n^{1/3}))$ lower bound on the threshold weight of a specific 2-decision list.

Sherstov's pattern-matrix method [30] showed how to take a function f satisfying the same condition required by Krause (i.e., $\widetilde{\deg}_{1-2^{-d}}(f) \geq d$), and turn it into a function F with discrepancy $2^{-\Omega(d)}$. By applying this transformation to the Minsky-Papert DNF or to OMB, Sherstov obtained a depth three circuit with discrepancy $\exp(-\Omega(n^{1/3}))$. Buhrman, Vereshchagin, and de Wolf independently proved an identical discrepancy bound via very different techniques [5]. These discrepancy bounds also implied corresponding lower bounds on Majority-of-Threshold Circuit Size, through standard transformations [23].

1.2.4 Recent Work on Approximating AC^0 Functions by Polynomials

A handful of recent works have established various forms of "hardness amplification" for approximate degree [28, 7, 8, 33, 28, 32, 19, 31]. Roughly speaking, these results show how to take a function g which is hard to approximate by degree d polynomials to error $1/3$, and

[7] Again, Podolskii describes his work in terms of degree-d threshold weight, but his results hold for approximate degree as well.

[8] In fact, Krause and Pudlák showed that F has threshold *length* 2^d, where threshold length is the least number of non-zero Fourier coefficients of any sign-representation for f. The threshold weight of f is always at least as large as its threshold length.

[9] Again, Krause phrased his lower bound in terms of the degree-d threshold weight of OMB, but his result is easily seen to imply the statement here.

turn g into a related function f that is hard to approximate by degree d polynomials to error exponentially close to 1. Specifically, in these works, f is obtained from g by block-composing g with another function h.

Let ED_M denote the well-known Element Distinctness function and $\overline{\mathrm{ED}}_M$ its negation. ED_M played a central role in recent works on hardness amplification for approximate degree [8, 33, 28] because, at the time these works were written, it exhibited the largest known approximate degree lower bound for any function in AC^0: $\widetilde{\deg}(\mathrm{ED}_M) = \tilde{\Omega}(M^{2/3})$ [1].[10]

Our prior work [8] showed that the function $f = \mathrm{OR}_N \circ \mathrm{ED}_M$ satisfies $\widetilde{\deg}_\varepsilon(f) \geq \tilde{\Omega}(M^{2/3})$ for $\varepsilon = 1 - 2^{-N}$, and used this result to obtain a depth three circuit such that $\mathcal{C}(F) = \exp(\tilde{\Omega}(n^{2/5}))$ for the three complexity measures $\mathcal{C}$ that we focus on in this work. Thaler [34] then showed that the function $f = \mathrm{OMB}_N \circ \overline{\mathrm{ED}}_M$ satisfies an identical lower bound, yielding another depth three circuit F (in fact, an $O(\log n)$-decision list) with $\mathcal{C}(F) = \exp(\tilde{\Omega}(n^{2/5}))$.

Sherstov [33] significantly strengthened the approach of [8] to obtain new *threshold degree* lower bounds for functions in AC^0. Specifically, in [33], for any $k \geq 2$, Sherstov exhibited a read-once formula of depth k (with polynomial bottom fan-in) that has threshold degree $\Omega(n^{\frac{k-1}{2k-1}})$. Applying the transformations of [17, 30, 16] to these circuits increases their depth by 1. In [28], Sherstov exhibited a depth four circuit of logarithmic bottom fan-in and threshold degree $\Omega(n^{1/2})$—applying the transformation of [17, 30, 16] to this circuit does not increase its depth, yielding a depth four circuit F satisfying $\mathcal{C}(F) = \exp(\Omega(n^{1/2}))$.

The $\exp(\Theta(\sqrt{n}))$ Barrier For Circuits of Depth Four. Recall that for each of the three complexity measures $\mathcal{C}$ in which we are interested, to construct a function of complexity at least 2^d under $\mathcal{C}$, it suffices to identify a function $f\colon \{-1,1\}^n \to \{-1,1\}$ such that

$$f \text{ cannot be approximated to error } 1 - 2^{-d} \text{ by polynomials of degree at most } d. \tag{1}$$

We now argue that for circuits of depth 4, prior techniques cannot accomplish this for $d \gg \sqrt{n}$, *even if they assume the existence of a DNF* $g\colon \{-1,1\}^M \to \{-1,1\}$ *with (one-sided) approximate degree* $\Omega(M)$.[11]

The methods of [8, 34] start with a function $g\colon \{-1,1\}^M \to \{-1,1\}$, and assume nothing about g other than that g has (one-sided) approximate degree at least d. They show how to turn g into a "harder" function $f = h \circ g$ by block-composing g with another function $h \in \{\mathrm{OR}_N, \mathrm{OMB}_N\}$. Quantitatively, the resulting bound is of the form $\widetilde{\deg}_{1-2^{-N}}(f) \geq d$. Clearly, one must set $N \geq d$ to obtain a bound of the form Eq. (1). Hence, even if g has the largest possible (one-sided) approximate degree, $d = M$, the best bound that can be obtained from the methods of [8, 34] is of the form $\widetilde{\deg}_{1-2^{-N}}(f) \geq N$, obtained by setting $M = N$. In this case, f is a function over $n = N^2$ variables, so these methods can only yield complexity bounds of the form $\exp(\Omega(N)) = \exp(\Omega(\sqrt{n}))$.

Both [8, 34] showed that their respective analyses are tight for many functions g. Hence, the $\exp(\sqrt{n})$ barrier is not merely an artifact of the analysis in these works.

---

[10] In very recent work [9], which was performed after the work described in the present manuscript, we have proved a nearly optimal lower bound on the approximate degree of AC^0: for any constant depth d, it exhibited a depth-d AC^0 circuit of approximate degree $\Omega(n^{1-2^{-\Omega(d)}})$). See Section 1.3 for details.

[11] One-sided approximate degree is a measure that is intermediate between approximate degree and threshold degree. One-sided approximate degree lower bounds is crucial to the analyses in [8, 33, 28]. However, we will not explicitly utilize one-sided approximate degree in our own results, so we do not formally define it here. While our work subsequent to this manuscript [9] has proved a nearly-linear lower bound on the approximate degree of AC^0 functions, the best known one-sided approximate degree lower bound for an AC^0 function currently remains $\tilde{\Omega}(M^{2/3})$, exhibited by ED_M [8].

Indeed, at least in the case of $h = \mathrm{OR}_N$, the barrier is inherent to any method that attempts to construct an f satisfying Eq. (1) by assuming nothing about a function $g\colon \{-1,1\}^M \to \{-1,1\}$ other than that g has (one-sided) approximate degree at least d, and then block-composing g with h. To see this, first observe that if $M \leq N$, then for *any* function $g\colon \{-1,1\}^M \to \{-1,1\}$, $(1/N)\sum_{i=1}^{N} g(x_i) + (N-1)/N$ is a polynomial of degree at most $M \leq n^{1/2}$ that approximates $f = \mathrm{OR}_N \circ g$ to error $1 - 1/N$.

Even if $M \geq N$, it is often the case that $\mathrm{OR}_N \circ g$ can be approximated to error $1 - 2^{-\tilde{O}(n^{1/2})}$ by a polynomial of degree $O(n^{1/2})$. Indeed, many functions g with large approximate degree (such as ED_M for example) can be approximated to error $1/3N^2$ by a *ratio* $q_1(x)/q_2(x)$ of two polynomials of logarithmic degree and weight quasi-polynomial in M and N. One can use q_1, q_2 to obtain a polynomial approximator p for $f = h \circ g$ such that $\deg(p) = O(N \log(M \cdot N))$, and p uniformly approximates f to error $1 - 2^{-O(N \cdot \mathrm{polylog}(M))}$. We omit the details for brevity, but the construction can be found in [4] (see also [28, Theorem 6.10]).

Sherstov [33, 28] introduced sophisticated and demanding methods that can prove stronger lower bounds for constant-depth circuits than [8, 34]. However, his methods apply block-composition multiple times, and crucially exploit alternation in the circuits computing the functions being composed; hence, in the context of Eq. (1), his analysis improves over [8, 34] only for circuits of greater depth or bottom fan-in than considered in those works. Furthermore, in [28, Section 9.4] Sherstov provides a detailed discussion on barriers facing his methods. In particular, he shows that the $\exp(\Omega(n^{1/2}))$ barrier is inherent to the class of functions he considers in [28], and is not an artifact of the analysis. He does indicate that his methods might be extendable to break the $\exp(\Omega(n^{1/2}))$ barrier by using circuits of depth 5 or greater.

1.3 Subsequent Work

Our recent works [9, 6], which were performed subsequent to the work described in this manuscript, introduced a very different method for performing hardness amplification in a manner that moves beyond block-composed functions. These works amplify the *degree* of the polynomials to which the lower bounds apply, thereby obtaining optimal or nearly optimal lower bounds for many functions in AC^0. In contrast, the techniques we introduce in the present manuscript amplify the *error* parameter for which the lower bounds apply. Combining the two methods to strengthen our lower bounds for depth three circuits, or circuits of larger (constant) depth, is an important direction for future work (see Section 3.3).

2 Preliminaries

Notation. We work with Boolean functions $f : \{-1,1\}^n \to \{-1,1\}$, where -1 corresponds to logical TRUE and $+1$ corresponds to logical FALSE. For a given Boolean function f, the function $\bar{f} := -f$ denotes its negation. The notation $[n]$ refers to the set $\{0, 1, \ldots, n\}$. For any $n \in \mathbb{N}$, fix a canonical injection $[n] \to \{-1,1\}^{\lceil \log(n+1) \rceil}$; we refer to the image of any $i \in [n]$ under this injection as the *binary representation* of i. All logarithms in this work are assumed to be taken in base 2.

Decision Lists and OMB. A k-decision list D of length L over the Boolean variables $x_1, \ldots, x_n$ is represented by a list of L pairs (C_0, b_0), (C_1, b_1), $\ldots$, (C_{L-1}, b_{L-1}) and a bit b_L where each C_i is a conjunction of at most k (possibly negated) variables, and each b_i is either -1 or 1. Given any $x \in \{-1,1\}^n$, the value of $D(x)$ is b_i if i is the smallest index such that C_i is made true by x; if no C_i is is true then $D(x) = b_L$.

Any k-decision list of length L is computed by a depth three circuit of size $O(L)$ and bottom fan-in $O(k)$. Indeed, letting $S = \{i \leq L : b_i = -1\}$, the circuit is

$$(b_L \wedge \bar{C}_0(x) \wedge \cdots \wedge \bar{C}_{L-1}(x)) \vee \bigvee_{i \in S} \left(C_i(x) \wedge \bar{C}_0(x) \wedge \cdots \wedge \bar{C}_{i-1}(x)\right).$$

To see that this is indeed a circuit of depth three with bottom fan-in $O(k)$, observe that for any conjunction C_i of width k, $\bar{C}_i$ is computed by a disjunction of width k.

Let $\mathrm{OMB} : \{-1,1\}^N \to \{-1,1\}$ denote a specific 1-decision list known as ODD-MAX-BIT, defined as follows. For $i = 1, 2, \ldots, N$, the conjunction $C_i(x) = x_{N-i}$ and $b_i = (-1)^{N-i}$. Finally, define $b_N = 1$. OMB can be equivalently defined in the following manner. On input $x = (x_1, \ldots, x_N)$, let $\beta(x)$ denote the largest index i such that $x_i = -1$, and let $\beta(x) = 0$ if no such index exists. Then

$$\mathrm{OMB}(x_1, \ldots, x_N) = \begin{cases} -1 \text{ if } \beta(x) \text{ is odd} \\ 1 \text{ otherwise.} \end{cases}$$

Beigel [3] showed that OMB has high approximate degree, even when the error parameter is exponentially close to 1. Specifically:

▶ **Theorem 3** (Beigel [3]). *There exists a constant $c > 0$ for which the following holds. Let $p(x)$ be a polynomial of degree at most d such that $p(x) \cdot \mathrm{OMB}_N(x) > 0$ for all $x \in \{-1,1\}^N$. Then there exists an $x \in \{-1,1\}^N$ such that $|p(x)| \geq 2^{cN/d^2} \cdot |p(1^N)|$. In particular, $\widetilde{deg}_\varepsilon(\mathrm{OMB}_N) \geq d$ for some $\varepsilon = 1 - 2^{-\Omega(N/d^2)}$.*

To prove Theorem 3, Beigel iteratively constructs a sequence of inputs $x^0, x^1, \ldots, x^{cN/d^2}$ for which $|p(x^{t+1})| \geq 2 \cdot |p(x^t)|$. He obtains these inputs by repeatedly applying the following lemma, which we will also make use of directly.

▶ **Lemma 4** (Beigel [3]). *Let $d, N \in \mathbb{N}$ and let $\ell \geq 10d^2$ such that N/ℓ is an integer. Consider the increasing family of sets $S_1 \subset S_2 \subset \cdots \subset S_{N/\ell} \subseteq \{-1,1\}^n$ defined by*

$$S_0 = \{1^N\}, S_1 = \{x : x_i = 1 \ \ \forall i > \ell\}, \ldots, S_t = \{x : x_i = 1 \ \forall i > t\ell\}, \ldots, S_{N/\ell} = \{-1,1\}^N.$$

Let $p(x)$ be a polynomial of degree at most d such that $p(x) \cdot \mathrm{OMB}_N(x) > 0$ for all $x \in S_{t+1} \setminus S_t$. Let $z \in S_t$. Then there exists a $z' \in S_{t+1} \setminus S_t$ such that $|p(z')| \geq 2 \cdot |p(z)|$.

3 Intuition and Discussion of Theorem 1

3.1 Overview of Our Function

As mentioned in Section 1, the function f that we exhibit in Theorem 1 is complicated to define. Hence, before formally defining f, we provide here some motivation for our definition. While the function we describe in this section differs from the f exhibited in Theorem 1, our description here highlights the main ideas underlying the construction of f itself.

Specifically, the function that we describe in this informal overview is a modification of the block-composed function $\mathrm{OMB}_N \circ \mathrm{OR}_M$. This is easily seen to be a sub-function of OMB_{2NM}, and in the formal statement and proof of Theorem 1, our construction relies on the function OMB_{2NM} directly. This is only for the sake of simplicity, and we remark that the proof of Theorem 1 carries over when one uses the function $\mathrm{OMB}_N \circ \mathrm{OR}_M$ instead. We choose to present this overview using the function $\mathrm{OMB}_N \circ \mathrm{OR}_M$ for two reasons. First, reasoning about this function gives the right intuition for both our lower bound and for the

approximating polynomial which it matches. Second, as discussed in Section 3.3, replacing the inner function OR_M with other functions of larger approximate degree yields natural candidates for improved lower bounds.

Recall that [34] showed that $\widetilde{\deg}_\varepsilon(\mathrm{OMB}_N \circ \overline{\mathrm{ED}}_M) = \tilde{\Omega}(M^{2/3})$ for $\varepsilon = 1 - 2^{-N}$. Moreover, this lower bound is essentially tight for $\mathrm{OMB}_N \circ \overline{\mathrm{ED}}_M$: there is in fact a polynomial p of degree $O(\log M)$ that approximates $\mathrm{OMB}_N \circ \overline{\mathrm{ED}}_M$ to error $\varepsilon = 1 - 2^{-O(N \log M)}$. The methods of [34] also show that $\widetilde{\deg}_\varepsilon(\mathrm{OMB}_N \circ \mathrm{OR}_M) = \Omega(M^{1/2})$ for $\varepsilon = 1 - 2^{-N}$, and there is a polynomial of degree $O(1)$ that approximates $\mathrm{OMB}_N \circ \mathrm{OR}_M$ to error $\varepsilon = 1 - 2^{-O(N \log M)}$.

Our goal is to modify $\mathrm{OMB}_N \circ \mathrm{OR}_M$ to obtain an $f : \{-1,1\}^n \to \{-1,1\}$ that is much harder to approximate by low-degree polynomials, while still ensuring that f is computed by an $O(\log n)$ decision list. Specifically, we will require, for some large constant k and small constant $\delta > 0$, $\widetilde{\deg}_\varepsilon(f) = \Omega(M^{1/2-\delta})$ for $\varepsilon = 1 - 2^{-N^k}$.

A natural first attempt to construct such an f is to block-compose $\mathrm{OMB}_N \circ \mathrm{OR}_M$ with the parity function on k variables. Specifically, let k be some constant, and consider the following function on $k \cdot N \cdot M$ variables: $\oplus_k \circ \mathrm{OMB}_N \circ \mathrm{OR}_M$, where $\oplus$ denotes the parity function. However, this function is still too easy to approximate: there is a polynomial of degree $O(k \log M)$ that approximates $\oplus_k \circ \mathrm{OMB}_N \circ \mathrm{OR}_M$ to error $1 - 2^{-O(kN \log M)}$. Indeed, letting p be the polynomial approximation to $\mathrm{OMB}_N \circ \mathrm{OR}_M$ described above, the polynomial $q(x_1, \ldots x_k) := 2^{-k} \prod_{i=1}^k p(x_i)$ does the trick.

We instead define f to be "just different enough" from $\oplus_k \circ \mathrm{OMB}_N \circ \mathrm{OR}_M$ to foil this construction of an approximating polynomial. Specifically, our f will first "pre-process" its input $(x_1, \ldots, x_k)$, before feeding it into $\oplus_k \circ \mathrm{OMB}_N \circ \mathrm{OR}_M$. The pre-processing step will introduce dependencies between blocks, so that an approximating polynomial for f will be unable to treat them independently in the manner of q.

In more detail, f will interpret its input x as k blocks, $x_1, \ldots, x_k$ (we will refer to $x_1, \ldots, x_k$ as "super-blocks", since each x_i will itself be interpreted as consisting of N blocks, which will themselves each be interpreted as consisting of M "sub-blocks"). For expository purposes, we focus in the remainder of this section on the case $k = 2$, so that there are only two super-blocks x_1, x_2 (The full construction is defined inductively, and described in the full version of this work). Assume for simplicity that $N + 1$ is a power of 2. The two super-blocks will not contain the same number of bits: x_1 will contain $N \cdot M$ bits, while x_2 will contain $N \cdot M \cdot \log(N+1)$ bits. We will ultimately treat x_1 as an input to $\mathrm{OMB}_N \circ \mathrm{OR}_M$; accordingly, let us interpret x_1 as consisting of N blocks, each containing M bits, so that we can write $x_1 = (x_{1,1}, \ldots, x_{1,N}) \in \left(\{-1,1\}^M\right)^N$. Let $\gamma(x_1) \in \{-1,1\}^{\log N}$ be the binary representation of the largest integer j satisfying $\mathrm{OR}(x_{1,j}) = -1$, and let $\gamma(x_1) = 0$ if no such j exists. That is, $\gamma(x_1)$ is the index of the "leading TRUE bit" that gets fed into OMB_N when evaluating $(\mathrm{OMB}_N \circ \mathrm{OR}_M)(x_1)$.

Similarly, we interpret x_2 as consisting of N blocks. However, each block now contains $M \log(N+1)$ bits, and is comprised of M sub-blocks, each consisting of $\log(N+1)$ bits. Let $\mathrm{EQ}_{\gamma(x_1)} : \{-1,1\}^{\log(N+1)} \to \{-1,1\}$ denote the function that outputs -1 if and only if its input equals $\gamma(x_1)$. Finally, let $u = (u_1, \ldots u_N) \in (\{-1,1\}^M)^N$ denote the vector obtained by applying $\mathrm{EQ}_{\gamma(x_1)}$ to each sub-block of x_2. That is, u is the vector obtained by first "pre-processing" each sub-block of x_2 with an "equality test" $\mathrm{EQ}_{\gamma(x_1)}$ that is determined by x_1. Finally, we define

$$f = ((\mathrm{OMB}_N \circ \mathrm{OR}_M)(x_1)) \oplus ((\mathrm{OMB}_N \circ \mathrm{OR}_M)(u)) \,. \tag{2}$$

Notice that the dependence of this pre-processing function on x_1 is actually quite mild: u only depends on the "leading TRUE bit" that gets fed into OMB_N when evaluating $\mathrm{OMB}_N \circ \mathrm{OR}_M(x_1)$. This mild dependence is what allows f to be computed by an $O(\log n)$ decision list.

It is not hard to see that an equivalent way to write f (that moreover helps reveal its structure as an $O(\log n)$-decision list) is:

$$\begin{aligned} f(x_1, x_2) = \mathrm{OMB}_{N^2+2N}(&\mathrm{OR}(u_1), \mathrm{OR}(u_2), \ldots, \mathrm{OR}(u_N), \\ &\mathrm{OR}(x_{1,1}), \mathrm{OR}(x_{1,1}) \wedge \mathrm{OR}(u_1), \mathrm{OR}(x_{1,1}) \wedge \mathrm{OR}(u_2), \ldots, \mathrm{OR}(x_{1,1}) \wedge \mathrm{OR}(u_N), \\ &\vdots \\ &\mathrm{OR}(x_{1,N}), \mathrm{OR}(x_{1,N}) \wedge \mathrm{OR}(u_1), \ldots, \mathrm{OR}(x_{1,N}) \wedge \mathrm{OR}(u_N)), \end{aligned} \tag{3}$$

where u is defined as above. It turns out that Representation (2) of f is useful for establishing lower bounds on the approximate degree of f, while Representation (3) is more useful for constructing approximating polynomials for f, and gaining intuition about f. In particular, Representation (3) suggests a natural method for approximating f: treat it as a 1-decision list over $N^2 + 2N$ "derived" variables (and then use an optimal method of approximating 1-decision lists, which are well-understood). The proof of our lower bound (Theorem 1) implicitly shows that this simple approach is essentially optimal. The next subsection briefly explains the details of this approximation method.

3.2 A Nearly Matching Upper Bound

We begin by giving the well-known sign-representing polynomial for OMB_N itself. Define $p\colon \{-1,1\}^N \to \mathbb{R}$ via $p(x_1, \ldots, x_N) := 1 + \sum_{i=1}^{N} (-2)^i \cdot (1 - x_i)/2$.

It is easy to see that $\mathrm{OMB}_N(x) \cdot p(x) > 0$ for all $x \in \{-1,1\}^N$, and in fact $2^{-N-1} \cdot p(x)$ approximates OMB_N to error $\varepsilon = 1 - 2^{-N-1}$. We now turn to constructing an approximant for the function $\mathrm{OMB}_N \circ \mathrm{OR}_M$. Our starting point is a polynomial q of degree $O(M^{1/2})$ satisfying the following two properties (cf. [34]).

$$q(x) = 0 \text{ for all } x \in \mathrm{OR}_M^{-1}(+1). \tag{4}$$

$$1 \leq q(x) \leq 2 \text{ for all } x \in \mathrm{OR}_M^{-1}(-1). \tag{5}$$

Denoting an $(N \cdot M)$-bit input as $(x_1, \ldots, x_N) \in \left(\{-1,1\}^M\right)^N$, it is easy to check that

$$\mathrm{OMB}_N \circ \mathrm{OR}_M(x_1, \ldots, x_N) = \mathrm{sgn}(g(x_1, \ldots, x_N)), \text{ where } g(x_1, \ldots, x_N) = 1 + \sum_{i=1}^{N} (-3)^i \cdot q(x_i).$$

In fact, $3^{-N-1} \cdot g(x)$ approximates $\mathrm{OMB}_N \circ \mathrm{OR}_M$ to error $1 - 3^{-N-1}$, and has degree equal to that of q. Recall (cf. Eq. (3)) that in the case $k = 2$, our function can be written as

$$\mathrm{OMB}_{N^2+2N}(\mathrm{OR}(u_1), \mathrm{OR}(u_2), \ldots, \mathrm{OR}(x_{1,N}) \wedge \mathrm{OR}(u_N)).$$

Using techniques similar to the above, one can obtain a degree $\tilde{O}(M^{1/2} \log N)$ polynomial that approximates this function to error $1 - 2^{-O(N^2)}$. Our lower bound shows this approximation is essentially optimal.

3.3 Prospects for Further Improved Lower Bounds

Fix a small constant $\delta > 0$ and large constant $\Gamma > 0$. A remarkable feature of the function f exhibited in Theorem 1 is that it simultaneously satisfies the following three properties:

- It has threshold degree $O(\log n)$.

- It has approximate degree $\tilde{\Theta}(n^{1/2})$.
- It has ε-approximate degree $\Omega(n^{1/2-\delta})$ for $\varepsilon = 1 - 2^{-n^{\Gamma}}$.

Hence, while f has very low threshold degree, it is essentially as hard to approximate f to error *super-exponentially* close to 1 (i.e., error as large as $1 - 2^{-n^{\Gamma}}$ for any constant $\Gamma > 0$), as it is to approximate to error $1/3$. To the best of our knowledge, ours is the first known function to exhibit these properties.

Clearly, improving the degree bound in Theorem 1 to be polynomially larger than $\Omega(n^{1/2})$ will require considering functions of approximate degree larger than $\Omega(n^{1/2})$. A natural approach is to simply replace the function OR_M appearing in the construction of Section 3.1 with a function of approximate degree polynomially larger than $\Omega(M^{1/2})$.

For example, for any constant depth $d > 0$, our recent work [9] identifies a depth-d AC^0 circuit F_d with approximate degree $\Omega(n^{1-2^{-\Omega(d)}})$. We conjecture that for any constant $\delta > 0$, replacing OR_M with F_d (for sufficiently large $d(\delta)$) in the construction of Section 3.1 yields a function of complexity $\exp(\Omega(n^{1-\delta}))$. This would imply a nearly optimal lower bound on the complexity of AC^0 under each of the complexity measures considered in this work.

4 Proof of Theorem 1

The proof of Theorem 1 is lengthy and technical, and deferred to the full version of this work. Here we consider an easier statement, the proof of which is much cleaner, while still capturing the main ideas of the general case.

4.1 Simplified Statement and its Proof

The function f that we exhibit in Theorem 1 is defined over k "superblocks", where k is an arbitrarily large constant (see Section 3.1 for motivation for the superblock terminology). Here, we consider the simpler case of exactly $k = 2$ superblocks.

Notation. Recall (cf. Section 2) that for $x = (x_1, \ldots, x_N) \in \{-1,1\}^N$, $\beta(x)$ denotes the largest index $i = 1, \ldots, N$ such that $x_i = -1$ (or $\beta(x) = 0$ if none exists). Assume for simplicity that $N+1$ is a power of two. Given an input $(x, y) \in \{-1,1\}^N \times (\{-1,1\}^{\log(N+1)})^N$, we interpret y as consisting of N blocks $y_1, \ldots, y_N$, each consisting of $\log(N+1)$ bits. Let $\mathrm{EQ}_{\beta(x)} \colon \{-1,1\}^{\log(N+1)} \to \{-1,1\}$ denote the function that outputs -1 if and only if its input equals the binary representation of $\beta(x)$. Finally, let $u = (u_1, \ldots u_N) \in \{-1,1\}^N$ denote the vector obtained by applying $\mathrm{EQ}_{\beta(x)}$ to each block of y. That is, u is the vector obtained by first "pre-processing" each block of y with an "equality test" $\mathrm{EQ}_{\beta(x)}$ that is determined by x.

Function Definition. Define F via:

$$F(x,y) = \mathrm{OMB}_N(x_1, \ldots, x_N) \oplus \mathrm{OMB}_N(u_1(x,y_1), \ldots, u_N(x,y_N)) \tag{6}$$

$$= \mathrm{OMB}_N(x_1, \ldots, x_N) \oplus \mathrm{OMB}_N(\mathrm{EQ}_{\beta(x)}(y_1), \ldots, \mathrm{EQ}_{\beta(x)}(y_N)). \tag{7}$$

▶ **Proposition 5.** *There exists a constant c for which the following holds. Let $d, n \in \mathbb{N}$ where $n = N + N\log(N+1)$. Let p be a polynomial of degree at most d such that $|p(x,y)| \geq 1$ and $p(x,y) \cdot F(x,y) > 0$ for all $(x,y) \in \{-1,1\}^n$. Then there exists an $(x,y) \in \{-1,1\}^n$ such that $|p(x,y)| \geq 2^{(cN/d^2)^2}$.*

To ease notation below, we will identify each block $y_i \in \{-1,1\}^{\log(N+1)}$ with the number in $[N]$ for which y_i is the binary representation. That is, while we will write each y_i as though it were a number $0, 1, \ldots, N$, it should always be thought of as the binary string representing that number.

Proof Idea. Let $p(x,y)$ be a polynomial of degree at most d that agrees with F in sign. Building on Beigel's proof of Theorem 3, we iteratively apply Lemma 4 to construct a sequence of inputs to the polynomial p, such that evaluating p on each point yields a value of (at least) twice the magnitude of the previous evaluation. By choosing these inputs carefully (and crucially exploiting the "pre-processing" step in the definition of F_2 that transforms y into the vector $u(x,y)$, before feeding it into OMB_N), we can apply Lemma 4 a total of $(cN/d^2)^2$ times. This is a quadratic improvement over the number of times Beigel is able to apply Lemma 4 to OMB_N itself. Podolskii [24] used related ideas to obtain a lower bound for a different function, but we are able to avoid the significant quantitative losses that are inherent to his approach.

In more detail, recall that Beigel's lower bound argument (cf. Theorem 3) for OMB_N started with the input $x^0 = 1^N$, and iteratively applied Lemma 4 to obtain inputs $x^1, \ldots, x^{cN/d^2}$ such that $|p(x^t)| \geq 2|p(x^{t-1})|$ for all $t \geq 1$. Roughly speaking, the first input to F that we construct is a point (x^1, y^0) such that $u(x^1, y^0) = 1^N$ and the last $N - 10d^2$ bits of x^1 are all set to 1. Since $u(x^1, y^0)$ is fed into OMB_N in the definition of F, we are able to apply Beigel's argument (Theorem 3) to obtain an input (x^1, y^1) such that $|p(x^1, y^1)| \geq 2^{cn/d^2} \cdot |p(x^1, y^0)|$. We then "use" the second block of $10d^2$ bits of the first superblock to "clean up" u, in the following sense: we find an x^2 whose last $N - 20d^2$ bits are all equal to 1, such that $u(x^2, y^1) = 1^N$ and $|p(x^2, y^1)| \geq |p(x^1, y^1)|$. This enables us to apply Beigel's argument (Theorem 3) a second time, finding an input (x^2, y^2) such that $|p(x^2, y^2)| \geq 2^{cn/d^2} \cdot |p(x^1, y^1)|$. We then use the third $10d^2$ bits of the first superblock to "clean up" u yet again, and repeat. We can continue the argument until we have "used up" all the bits of the first superblock, at which point we have obtained the desired lower bound.

Proof of Proposition 5. Let $\ell = 10d^2$ and consider the increasing family of sets $S_0 \subset S_1 \subset \cdots \subset S_{N/\ell} \subseteq \{-1,1\}^N$ defined as in Lemma 4. Let p be a polynomial of degree at most d such that $p(x,y) \cdot F(x,y) > 0$ for all $(x,y) \in \{-1,1\}^n$. We iteratively construct a sequence of inputs $(x^0, y^0), (x^1, y^1), \ldots, (x^{N/\ell}, y^{N/\ell})$ to p such that:

- Each $x^t \in S_t$ and each $y^t \in [t\ell]^N$,
- $|p(x^0, y^0)| \geq 1$, and
- $|p(x^{t+1}, y^{t+1})| \geq 2^{cN/d^2} \cdot |p(x^t, y^t)|$ for each $t = 0, \ldots, N/\ell - 1$.

At the conclusion of this process, we obtain an input $(x^{N/\ell}, y^{N/\ell})$ such that $|p(x^{N/\ell}, y^{N/\ell})| \geq 2^{(cN/d^2)^2}$.

We may take as the first input (x^0, y^0) the point $(1^N, 0^N)$. We construct the remaining inputs (x^t, y^t) iteratively. The following claim formalizes this iterative process.

▶ **Claim 6.** *Let p be a polynomial of degree at most d and suppose $p(x,y) \cdot F(x,y) > 0$ for all $(x,y) \in \{-1,1\}^n$. Let (x^t, y^t) be an input with $x^t \in S_t$ and $y^t \in [t\ell]^N$. Then there exists an input (x^{t+1}, y^{t+1}) such that $|p(x^{t+1}, y^{t+1})| \geq 2^{cN/d^2} \cdot |p(x^t, y^t)|$, where $x^{t+1} \in S_{t+1}$ and $y^{t+1} \in [(t+1)\ell]^N$.*

Proof. We prove the claim in two steps. First, we show that there exists an $x^{t+1} \in S_{t+1}$ for which $|p(x^{t+1}, y^t)| \geq |p(x^t, y^t)|$. Second, we show that there exists a $y^{t+1} \in [(t+1)\ell]^N$ such that $|p(x^{t+1}, y^{t+1})| \geq 2^{cN/d^2} \cdot |p(x^{t+1}, y^t)|$. Putting these steps together yields $|p(x^{t+1}, y^{t+1})| \geq 2^{cN/d^2} \cdot |p(x^t, y^t)|$.

Step 1. We examine the function $F(x, y^t)$ (viewed as a function only of x). By construction, each block $y_i^t \leq t\ell$. Thus, $\mathrm{EQ}_{\beta(x)}(y_i^t) = 1$ for all $x \in S_{t+1} \setminus S_t$, and hence

$$\mathrm{OMB}_N(\mathrm{EQ}_{\beta(x)}(y_1^t), \ldots, \mathrm{EQ}_{\beta(x)}(y_N^t)) = \mathrm{OMB}_N(1^N) = 1$$

for all such inputs. As a result, $F(x, y^t) = \mathrm{OMB}_N(x)$ whenever $x \in S_{t+1} \setminus S_t$.

Now consider the polynomial $q : \{-1, 1\}^N \to \mathbb{R}$ defined by $q(x) = p(x, y^t)$. Then $q(x) \cdot \mathrm{OMB}_N(x) > 0$ for all $x \in S_{t+1} \setminus S_t$. By assumption, $x^t \in S_t$. Thus, by Lemma 4, there exists an $x^{t+1} \in S_{t+1}$ such that $|q(x^{t+1})| \geq 2 \cdot |q(x^t)|$. Unpacking the definition of q, we see that in particular, $|p(x^{t+1}, y^t)| \geq |p(x^t, y^t)|$.

Step 2. We now show that there exists a $y^{t+1} \in [(t+1)\ell]^N$, such that $|p(x^{t+1}, y^{t+1})| \geq 2^{cN/d^2} \cdot |p(x^{t+1}, y^t)|$. For $w \in \{-1, 1\}^N$, define the string y_w by $(y_w)_i = \beta(x^{t+1})$ if $w_i = -1$ and $(y_w)_i = y_i^t$ if $w_i = 1$. Note that since $\beta(x^{t+1}) \leq (t+1)\ell$ and each $y_i^t \in [t\ell]$, we have that $y_w \in [(t+1)\ell]^N$ for every $w \in \{-1, 1\}^N$. Consider the function $E : \{-1, 1\}^N \to \{-1, 1\}$ defined by $E(w) := F(x^{t+1}, y_w)$, and observe that

$$E(w) = \mathrm{OMB}_N(x^{t+1}) \oplus \mathrm{OMB}_N(w).$$

Now consider the polynomial $r(w) := p(x^{t+1}, y_w)$. Observe that y_w is an affine function of w, i.e., we can write

$$r(w) = p\left(x^{t+1}, \left(\frac{1-w_1}{2}\right) \cdot \beta(x^{t+1}) + \left(\frac{1+w_1}{2}\right) \cdot y_1^t, \ldots, \left(\frac{1-w_N}{2}\right) \cdot \beta(x^{t+1}) + \left(\frac{1+w_N}{2}\right) \cdot y_N^t\right).$$

Thus r is a polynomial with $\deg r \leq \deg p \leq d$. Moreover, $r(w) \cdot E(w) > 0$ for all $w \in \{-1, 1\}^N$. Since E is either the function OMB_N or its negation, we conclude by Theorem 3 that there exists a w^* such that $|r(w^*)| \geq 2^{cN/d^2} \cdot |r(1^N)|$. Setting $y^{t+1} := y_{w^*}$ thus yields

$$|p(x^{t+1}, y^{t+1})| = |r(w^*)| \geq 2^{cN/d^2} \cdot |r(1^N)| = 2^{cN/d^2} \cdot |p(x^{t+1}, y^t)|,$$

as we wanted to show. ◀

With Claim 6 established, we conclude the proof of Proposition 5. ◀

References

1. Scott Aaronson and Yaoyun Shi. Quantum lower bounds for the collision and the element distinctness problems. *J. ACM*, 51(4):595–605, 2004. doi:10.1145/1008731.1008735.
2. 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. IEEE Computer Society, 1986. doi:10.1109/SFCS.1986.15.
3. Richard Beigel. Perceptrons, PP, and the Polynomial Hierarchy. *Computational Complexity*, 4:339–349, 1994. doi:10.1007/BF01263422.
4. Richard Beigel, Nick Reingold, and Daniel A. Spielman. PP is closed under intersection. *J. Comput. Syst. Sci.*, 50(2):191–202, 1995. doi:10.1006/jcss.1995.1017.
5. Harry Buhrman, Nikolai K. Vereshchagin, and Ronald de Wolf. On computation and communication with small bias. In *22nd Annual IEEE Conference on Computational Complexity (CCC 2007), 13-16 June 2007, San Diego, California, USA*, pages 24–32. IEEE Computer Society, 2007. doi:10.1109/CCC.2007.18.

6 Mark Bun, Robin Kothari, and Justin Thaler. The polynomial method strikes back: Tight quantum query bounds via dual polynomials. In *Symposium on Theory of Computing (STOC)*, pages 297–310. ACM, 2018. doi:10.1145/3188745.3188784.

7 Mark Bun and Justin Thaler. Dual lower bounds for approximate degree and markov-bernstein inequalities. In Fedor V. Fomin, Rusins Freivalds, Marta Z. Kwiatkowska, and David Peleg, editors, *ICALP (1)*, volume 7965 of *Lecture Notes in Computer Science*, pages 303–314. Springer, 2013. doi:10.1007/978-3-642-39206-1_26.

8 Mark Bun and Justin Thaler. Hardness amplification and the approximate degree of constant-depth circuits. 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 I*, volume 9134 of *Lecture Notes in Computer Science*, pages 268–280. Springer, 2015. Full version available at http://eccc.hpi-web.de/report/2013/151. doi:10.1007/978-3-662-47672-7_22.

9 Mark Bun and Justin Thaler. A nearly optimal lower bound on the approximate degree of AC^0. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 1–12, 2017. doi:10.1109/FOCS.2017.10.

10 Mikael Goldmann, Johan Håstad, and Alexander A. Razborov. Majority gates VS. general weighted threshold gates. *Computational Complexity*, 2:277–300, 1992. doi:10.1007/BF01200426.

11 Mika Göös, Toniann Pitassi, and Thomas Watson. The landscape of communication complexity classes. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, pages 86:1–86:15, 2016. doi:10.4230/LIPIcs.ICALP.2016.86.

12 András Hajnal, Wolfgang Maass, Pavel Pudlák, Mario Szegedy, and György Turán. Threshold circuits of bounded depth. *J. Comput. Syst. Sci.*, 46(2):129–154, 1993. doi:10.1016/0022-0000(93)90001-D.

13 Hartmut Klauck. Lower bounds for quantum communication complexity. *SIAM J. Comput.*, 37(1):20–46, 2007. doi:10.1137/S0097539702405620.

14 Adam R. Klivans and Rocco A. Servedio. Learning DNF in time $2^{\tilde{O}(n^{1/3})}$. *J. Comput. Syst. Sci.*, 68(2):303–318, 2004. doi:10.1016/j.jcss.2003.07.007.

15 Adam R. Klivans and Rocco A. Servedio. Toward attribute efficient learning of decision lists and parities. *Journal of Machine Learning Research*, 7:587–602, 2006. URL: http://www.jmlr.org/papers/v7/klivans06a.html.

16 Matthias Krause. On the computational power of boolean decision lists. *Computational Complexity*, 14(4):362–375, 2006. doi:10.1007/s00037-005-0203-0.

17 Matthias Krause and Pavel Pudlák. On the computational power of depth-2 circuits with threshold and modulo gates. *Theor. Comput. Sci.*, 174(1-2):137–156, 1997. doi:10.1016/S0304-3975(96)00019-9.

18 Matthias Krause and Pavel Pudlák. Computing boolean functions by polynomials and threshold circuits. *Computational Complexity*, 7(4):346–370, 1998. doi:10.1007/s000370050015.

19 Troy Lee. A note on the sign degree of formulas. *arXiv*, abs/0909.4607, 2009. arXiv:0909.4607.

20 Nati Linial and Adi Shraibman. Learning complexity vs. communication complexity. In *Proceedings of the 23rd Annual IEEE Conference on Computational Complexity, CCC 2008, 23-26 June 2008, College Park, Maryland, USA*, pages 53–63. IEEE Computer Society, 2008. doi:10.1109/CCC.2008.28.

21 Nick Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2(4):285–318, 1987. doi:10.1007/BF00116827.

22 Marvin Minsky and Seymour Papert. *Perceptrons - an introduction to computational geometry*. MIT Press, 1969.

23 Noam Nisan. The communication complexity of threshold gates. In *Combinatorics, Paul Erdos is Eighty*, pages 301–315, 1994.

24 Vladimir V. Podolskii. A uniform lower bound on weights of perceptrons. In Edward A. Hirsch, Alexander A. Razborov, Alexei L. Semenov, and Anatol Slissenko, editors, *Computer Science - Theory and Applications, Third International Computer Science Symposium in Russia, CSR 2008, Moscow, Russia, June 7-12, 2008, Proceedings*, volume 5010 of *Lecture Notes in Computer Science*, pages 261–272. Springer, 2008. doi:10.1007/978-3-540-79709-8_27.

25 Vladimir Vladimirovich Podolskii. Perceptrons of large weight. *Problems of Information Transmission*, 45(1):46–53, 2009.

26 Ronald L. Rivest. Learning decision lists. *Machine Learning*, 2(3):229–246, 1987. doi:10.1007/BF00058680.

27 Rocco A. Servedio, Li-Yang Tan, and Justin Thaler. Attribute-efficient learning and weight-degree tradeoffs for polynomial threshold functions. In Shie Mannor, Nathan Srebro, and Robert C. Williamson, editors, *COLT*, volume 23 of *JMLR Proceedings*, pages 14.1–14.19. JMLR.org, 2012. URL: http://www.jmlr.org/proceedings/papers/v23/servedio12/servedio12.pdf.

28 A. A. Sherstov. The power of asymmetry in constant-depth circuits. In *FOCS*, 2015. Full version available at http://eccc.hpi-web.de/report/2015/147/.

29 Alexander A. Sherstov. Separating AC^0 from depth-2 majority circuits. *SIAM J. Comput.*, 38(6):2113–2129, 2009. doi:10.1137/08071421X.

30 Alexander A. Sherstov. The pattern matrix method. *SIAM J. Comput.*, 40(6):1969–2000, 2011. doi:10.1137/080733644.

31 Alexander A. Sherstov. Approximating the and-or tree. *Theory of Computing*, 9(20):653–663, 2013.

32 Alexander A. Sherstov. The intersection of two halfspaces has high threshold degree. *SIAM J. Comput.*, 42(6):2329–2374, 2013. doi:10.1137/100785260.

33 Alexander A. Sherstov. Breaking the Minsky-Papert barrier for constant-depth circuits. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 223–232. ACM, 2014. doi:10.1145/2591796.2591871.

34 Justin Thaler. Lower bounds for the approximate degree of block-composed functions. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, pages 17:1–17:15, 2016. doi:10.4230/LIPIcs.ICALP.2016.17.

35 Leslie G. Valiant. A theory of the learnable. In Richard A. DeMillo, editor, *Proceedings of the 16th Annual ACM Symposium on Theory of Computing, April 30 - May 2, 1984, Washington, DC, USA*, pages 436–445. ACM, 1984. doi:10.1145/800057.808710.

A Applications

A.1 An Application in Communication Complexity

We briefly describe an additional application of our results. By combining Theorem 1 with standard machinery, we obtain an improved separation between the analogues of the complexity classes $\mathbf{PP}$ and $\mathbf{P^{NP}}$ in communication and query complexity. Specifically, for a function $F\colon \{-1,1\}^n \times \{-1,1\}^n \to \{-1,1\}$, let $\mathbf{PP}(F)$ and $\mathbf{P^{NP}}(F)$ respectively denote the least cost of a $\mathbf{PP}$ and $\mathbf{P^{NP}}$ communication protocol for F. In both the

communication and query complexity settings, for any constant $\delta > 0$, we exhibit an F satisfying $\mathbf{PP}(F) = \Omega(n^{1/2-\delta})$ and $\mathbf{P}^{\mathbf{NP}}(F) = O(\log^2 n)$. This improves over prior work that gave an F satisfying $\mathbf{PP}(F) = \tilde{\Omega}(n^{2/5})$ and $\mathbf{P}^{\mathbf{NP}}(F) = O(\log n)$ [34] and earlier work of Buhrman et al. that gave an F satisfying $\mathbf{PP}(F) = \Omega(n^{1/3})$ and $\mathbf{P}^{\mathbf{NP}}(F) = O(\log n)$ [5]. We direct the interested reader to [34] for further details on this application.

A.2 An Application to Learning Theory

A notorious open question in learning theory, dating back to Valiant's seminal paper introducing the PAC model [35], is whether polynomial size DNF can be PAC learned in polynomial time. The fastest known algorithm, due to Klivans and Servedio [14] runs in time $\exp(\tilde{O}(n^{1/3}))$, and follows from their (tight) $\tilde{O}(n^{1/3})$ upper bound on the threshold degree of any polynomial size DNF. Klivans and Servedio's threshold degree upper bound makes use of polynomials with coefficients that are *doubly*-exponentially large (as large as $2^{2^{\tilde{\Theta}(n^{1/3})}}$). They pose the following open question:

Open Question (Klivans and Servedio [14]). Is every polynomial size DNF computed by a polynomial threshold function with degree $\tilde{O}(n^{1/3})$ and coefficients of magnitude at most $2^{\tilde{O}(n^{1/3})}$?

A positive resolution of this open question would imply much *simpler* (though not necessarily faster) learning algorithms for DNFs. Our results can be viewed as a significant step toward resolving Klivans and Servedio's question in the *negative* (in fact, achieving a full resolution of the open question was our original motivation for this work). This is because Theorem 1 implies that for any constants $\Gamma, \delta > 0$ there is an $O(\log n)$-decision list F of polynomial length, such that any polynomial of degree $O(n^{1/2-\delta})$ that sign-represents F requires coefficients of magnitude $2^{n^{\Gamma}}$. While $O(\log n)$-decision lists are not necessarily computed by DNFs, they seem to be only slightly more expressive than DNFs. In particular, they are computed by depth 3 circuits of logarithmic bottom fan-in.

Furthermore, our result points out an inherent limitation of Klivans and Servedio's construction. Their threshold degree upper bound for DNFs works as follows. They first transform the DNF into a (slight generalization of) a k-decision list, for some $k = O(n^{1/3})$. Second, they prove a $\tilde{O}(k)$ upper bound on the threshold degree of any k-decision list. Our result implies that the super-exponentially large coefficients arising in the second step of Klivans and Servedio's construction are necessary, even for k bounded by $O(\log n)$. Hence, any affirmative resolution of Klivans and Servedio's question cannot go through k-decision lists, even for $k = O(\log n)$.

A.3 Threshold Weight of Depth Three Circuits

A *polynomial threshold function* (PTF) for a Boolean function $f : \{-1,1\}^n \to \{-1,1\}$ is a polynomial $p : \{-1,1\}^n \to \mathbb{R}$ with *integer coefficients* that agrees in sign with f on all Boolean inputs. The *weight* of an n-variate polynomial p is the sum of the absolute values of its coefficients. The *degree-d threshold weight* of a Boolean function $f : \{-1,1\}^n \to \{-1,1\}$, denoted $W(f,d)$, is defined to be the least weight of a degree-d PTF for f. We let $W(f)$ denote the quantity $W(f,n)$, i.e., the least weight of any threshold function for f regardless of its degree. Threshold weight upper bounds underly some of the most powerful techniques in computational learning theory based on the classic Perceptron [22] and Winnow [21] algorithms (see [8, Section 8.3] for a discussion). Thus, our threshold weight lower bounds impose limitations on how efficiently such algorithms can learn depth three circuits.

Degree-d threshold weight is closely related to ε-approximate degree when ε is very close to 1:

▶ **Lemma 7.** *Let $f : \{-1,1\}^n \to \{-1,1\}$ be a Boolean function, and let $w > 0$. If $\widetilde{deg}_{1-\frac{1}{w}}(f) > d$, then $W(f,d) > w$.*

Proof. We prove the contrapositive, i.e., that any PTF p for f having weight w and degree d can be transformed into a uniform approximation to f with error $1 - \frac{1}{w}$. Let p be such a PTF. Since p has integer coefficients and is nonzero on Boolean inputs, $|p(x)| \geq 1$ on $\{-1,1\}^n$. Moreover, $|p(x)| \leq w$ by the weight bound, so the polynomial $\frac{1}{w} \cdot p(x)$ satisfies $|\frac{1}{w} \cdot p(x) - f(x)| \leq 1 - \frac{1}{w}$ for every $x \in \{-1,1\}^n$. ◀

Thus, our main results yield new lower bounds on the degree-d threshold weight of circuits of depth three.

▶ **Corollary 8.** *For any arbitrarily small constant $\delta > 0$ and any arbitrarily large constant $\Gamma > 1$, there exists a depth three Boolean circuit $f\colon \{-1,1\}^n \to \{-1,1\}$ (with logarithmic bottom fan-in) such that $W(f, n^{1/2-\delta}) > 2^{n^\Gamma}$.*

Moreover, a result of Krause [16] allows us to translate each of these lower bounds into a *degree independent* threshold weight lower bound for a related function.

▶ **Lemma 9** ([16], Lemma 3.4)**.** *Let $f : \{-1,1\}^n \to \{-1,1\}$ be a Boolean function, and define $F : \{-1,1\}^{3n} \to \{-1,1\}$ by*

$$F(x_1,\dots,x_n,y_1,\dots,y_n,z_1,\dots,z_n) := f(\dots,(\bar{z}_i \wedge x_i) \vee (z_i \wedge y_i),\dots).$$

Then $W(F) \geq W(f,d)$ for all d for which $2^d \geq W(f,d)$.

When this transformation is applied to a function f computed by a Boolean circuit of depth d with logarithmic bottom fan-in, the resulting function F is also computed by a depth d circuit with logarithmic bottom fan-in. To see this, note that if g is any function that depends on $O(\log n)$ variables, then $G(x,y,z) := g((\bar{z} \wedge x) \vee (z \wedge y))$ also depends on $O(\log n)$ variables. Hence, G is computed by either a DNF or CNF of size $\mathrm{poly}(n)$ and bottom fan-in $O(\log n)$. So while F is naturally computed by a circuit of depth $d+2$, the bottom three levels of gates can be replaced by such DNF or CNF formulae so as to merge a layer of gates and obtain a depth d circuit with logarithmic bottom fan-in.

▶ **Corollary 10.** *For any arbitrarily small constant $\delta > 0$, there exists a depth three Boolean circuit $F\colon \{-1,1\}^n \to \{-1,1\}$ (with logarithmic bottom fan-in) such that $W(F) > \exp(\Omega(n^{1/2-\delta}))$.*

While the weight bounds of Corollaries 8 and 10 are stated for polynomial threshold functions over $\{-1,1\}^n$ (i.e., for polynomials that are integer linear combinations of parities), a now standard transformation [18] shows that the same threshold weight lower bound also holds for polynomials over $\{0,1\}^n$ (i.e., for integer linear combinations of conjunctions) up to polynomial factors.

A.4 Discrepancy of Depth Three Circuits

Discrepancy is a central quantity in communication complexity and circuit complexity. For instance, an upper bound on the discrepancy of a Boolean function $f : X \times Y \to \{-1,1\}$ yields lower bounds for computing f in essentially every model of communication complexity.

In particular, the discrepancy of f essentially characterizes it's small-bias communication complexity in the **PP** model of Babai et al. [2]. Theorem 1 yields a new exponentially small upper bound on the discrepancy of a depth three circuit.

For a Boolean function $f : X \times Y \to \{-1, 1\}$, let $M^{(f)}$ be its communication matrix $M^{(f)} = [f(x, y)]_{x \in X, y \in Y}$. A combinatorial rectangle of $X \times Y$ is a set of the form $A \times B$ with $A \subseteq X$ and $B \subseteq Y$. For a distribution μ over $X \times Y$, the discrepancy of f with respect to μ is defined to be the maximum over all rectangles R of the *bias* of f on R. That is:

$$\operatorname{disc}_\mu(f) = \max_R \left| \sum_{(x,y) \in R} \mu(x, y) f(x, y) \right|.$$

The discrepancy of f, denoted $\operatorname{disc}(f)$, is defined to be $\min_\mu \operatorname{disc}_\mu(f)$.

Sherstov's pattern matrix method [30] shows how to generically transform an AC^0 function with high threshold degree or high threshold weight into another AC^0 function with low discrepancy.

▶ **Theorem 11** (cf. [30], adapted from Corollary 1.2 and Theorem 7.3)**.** *Let $f : \{-1, 1\}^n \to \{-1, 1\}$ be given, and define the communication problem $F : \{-1, 1\}^{4n} \times \{-1, 1\}^{4n} \to \{-1, 1\}$ by*

$$F(x, y) = f(\ldots, \vee_{j=1}^4 (x_{i,j} \wedge y_{i,j}), \ldots).$$

Then for every integer $d \geq 0$, we have

$$\operatorname{disc}(F)^2 \leq \max \left\{ \frac{2n}{W(f, d-1)}, 2^{-d} \right\}.$$

Recall that $W(f, d-1)$ is the least weight of any degree $d-1$ PTF for f. We apply the pattern matrix method to the function f of Corollary 8. By the same argument as in Section A.3, the pattern matrix method does not increase the depth of the circuits computing these functions. We thus obtain a new discrepancy upper bound for circuits of depth three:

▶ **Corollary 12.** *For any arbitrarily small constant $\delta > 0$, there exists a depth three Boolean circuit $F \colon \{-1, 1\}^n \to \{-1, 1\}$ (with logarithmic bottom fan-in) such that* $\operatorname{disc}(F) < \exp(-\Omega(n^{1/2-\delta}))$.

Application to Circuit Complexity. It is well-known that a discrepancy upper bound for a function F yields a lower bound on the size of Majority-of-Threshold circuits computing F [10, 12, 23, 29]. Indeed, the exponential Majority-of-Threshold circuit size lower bounds of [29, 30, 5, 8, 33, 28] for AC^0 are all proved using discrepancy. Our discrepancy upper bound of Corollary 12 sharpens these previous lower bounds by yielding a depth three Boolean circuit F of polynomial size such that any Majority-of-Threshold circuit computing F requires size $\exp\left(\Omega(n^{1/2-\delta})\right)$.

▶ **Corollary 13.** *For any arbitrarily small constant $\delta > 0$, there exists a depth three Boolean circuit $F \colon \{-1, 1\}^n \to \{-1, 1\}$ (with logarithmic bottom fan-in) such that any Majority-of-Threshold circuit computing F has size at least* $\exp(\Omega(n^{1/2-\delta}))$.

Combining Corollaries 10, 12, and 13 yields Corollary 2 from the introduction, noting that the function exhibited in Corollary 10 appears as a subfunction of the one in Corollaries 12 and 13.

Speeding up Switch Markov Chains for Sampling Bipartite Graphs with Given Degree Sequence

Corrie Jacobien Carstens
Korteweg-de Vries Institute for Mathematics, Amsterdam, The Netherlands
c.j.carstens@uva.nl

Pieter Kleer
Centrum Wiskunde & Informatica (CWI), Amsterdam, The Netherlands
kleer@cwi.nl

Abstract

We consider the well-studied problem of uniformly sampling (bipartite) graphs with a given degree sequence, or equivalently, the uniform sampling of binary matrices with fixed row and column sums. In particular, we focus on Markov Chain Monte Carlo (MCMC) approaches, which proceed by making small changes that preserve the degree sequence to a given graph. Such Markov chains converge to the uniform distribution, but the challenge is to show that they do so quickly, i.e., that they are *rapidly mixing*.

The standard example of this Markov chain approach for sampling bipartite graphs is the switch algorithm, that proceeds by locally switching two edges while preserving the degree sequence. The Curveball algorithm is a variation on this approach in which essentially multiple switches (trades) are performed simultaneously, with the goal of speeding up switch-based algorithms. Even though the Curveball algorithm is expected to mix faster than switch-based algorithms for many degree sequences, nothing is currently known about its mixing time. On the other hand, the switch algorithm has been proven to be rapidly mixing for several classes of degree sequences.

In this work we present the first results regarding the mixing time of the Curveball algorithm. We give a theoretical comparison between the switch and Curveball algorithms in terms of their underlying Markov chains. As our main result, we show that the Curveball chain is rapidly mixing whenever a switch-based chain is rapidly mixing. We do this using a novel state space graph decomposition of the switch chain into Johnson graphs. This decomposition is of independent interest.

2012 ACM Subject Classification Theory of computation → Random walks and Markov chains

Keywords and phrases Binary matrix, graph sampling, Curveball, switch, Markov chain decomposition, Johnson graph

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.36

Related Version A full version of this work is available at [5], `https://arxiv.org/abs/1709.07290`.

Funding This work is supported by NWO Gravitation Project NETWORKS, Grant Number 024.002.003.

Acknowledgements We are grateful to Annabell Berger and Catherine Greenhill for some useful discussions and comments on an earlier version of this work. We also thank the anonymous reviewers for their valuable comments.

1 Introduction

The problem of sampling bipartite graphs with a given degree sequence, or equivalently the sampling of binary matrices with fixed row and column sums (marginals) has received a lot of attention in theoretical research, in particular Markov chain Monte Carlo switch-based approaches, see, e.g., [23, 6, 18, 20, 25, 14, 17]. Furthermore, it has many applications, for instance in network science [26] and in the field of ecology [28, 30] where it is used as a null-model. Hence, it is important that the sampling is unbiased and fast. In the Markovian approach, one repeatedly applies small changes to a given binary matrix, that represents the adjacency matrix of a bipartite graph in the natural way, with each change preserving the marginals (the bipartite degree sequence). The idea is that after a sufficient number of changes, i.e., transitions of the underlying Markov chain, the resulting matrix corresponds to an approximately uniform sample from the set of all binary matrices having the given marginals. The number of steps needed to get within a given distance of the uniform distribution is known as the *mixing time* of the Markov chain, and the chain is said to be rapidly mixing if this number can be bounded by a polynomial in the size of the bipartitions considered (the parameters m and n as introduced in Section 4).

The best-known probabilistic procedure for making these small changes uses *switches*, in which two one-entries of the matrix are 'exchanged' with two zero-entries of the matrix see, e.g., [28]. The resulting Markov chain is called the switch chain. A possible implementation for such a chain was proposed by Kannan, Tetali and Vempala [23]. One first chooses two rows and columns uniformly at random, and, if the 2×2 submatrix formed by these rows and columns corresponds to a *checkerboard*, the zero and one entries are exchanged, see Section 4.1 for details. The Curveball algorithm was introduced in [32] and rediscovered in [30]. Both articles focus on applications and are motivated by the need to speed up switch-based algorithms. In the Curveball algorithm one again chooses two rows uniformly at random, but then continues by randomly exchanging all one and zero entries on the chosen rows in a way that preserves the marginals. Such an exchange is called a *(binomial) trade*. A detailed description is given in Section 4.2.

The main result of this work is a theoretical comparison of the relaxation times of the Kannan-Tetali-Vempala (KTV) switch chain, and the Curveball chain. The relaxation time of a Markov chain essentially determines its mixing time. In particular, our result implies that the Curveball chain is rapidly mixing whenever the KTV switch chain is rapidly mixing, but that there is at most a quadratic improvement in relaxation time (Theorem 3). We prove this statement in the more general setting in which there is a set of forbidden entries, that have to be zero. This allows us to also compare the chains for the sampling of a simple directed graph with given degree sequence also, as its adjacency matrix can be given by a square binary matrix with zeros on the diagonal.

In order to establish our results, we introduce a general comparison framework inspired by, and based on, the notion of a heat-bath Markov chain, using the definition by Dyer, Greenhill and Ullrich [12]. This framework essentially compares a given Markov chain with a *locally refined* version, which we will call its *heat-bath variant*. We introduce a novel decomposition of the state space of the switch and Curveball chain in order to apply this framework. In particular, this decomposition allows us to show that the transition matrix of the KTV-switch chain only has non-negative eigenvalues. This result is of independent

interest, as it implies that the chain does not have to be made lazy.[1] Making a Markov chain lazy is an easy way to guarantee that its transition matrix only has non-negative eigenvalues, but the procedure has been described, e.g., as 'unnatural' (see the note of Greenhill [19] for a discussion and references).

As an additional application of our framework, we show it can be used to prove that the parallel Curveball chain [4] is rapidly mixing whenever the Curveball chain is rapidly mixing. In the parallel Curveball chain multiple independent binomial trades are performed in parallel.

Some proofs are omitted from the main text, but can be found in Appendix A.

1.1 Related work

We refer the reader to [17] for a nice exposition on the switch Markov chain. We give a brief overview. Kannan, Tetali and Vempala [23] conjectured that the KTV-switch chain is rapidly mixing for all fixed row and column sums. Miklós, Erdős and Soukup [25] proved the conjecture for half-regular binary matrices, in which all the row sums are equal (or all column sums), and Erdős, Kiss, Miklós and Soukup [14] extended this result to almost half-regular marginals. The authors of [14] prove this in a slightly more general context where there might be certain forbidden edge sets. See also [17, 15] for more results. For rapid mixing results of the switch chain for general (un)directed graphs, see also, e.g., Cooper, Dyer and Greenhill [6], Greenhill [18], and Greenhill and Sfragara [21]. In general, the switch chain can be slow mixing in the presence of forbidden edges sets, which follows from the work of Bezáková, Bhatnagar and Randall [2]. The Curveball algorithm was first described by Verhelst [32] and a slightly different version was later independently formulated by Strona, Nappo, Boccacci, Fattorini and San-Miguel-Ayanz [30]. The name Curveball algorithm was introduced in [30]. The Curveball chain has also been formulated for (un)directed graphs, see Carstens, Berger and Strona [4].

Showing rapid mixing for the switch chain has proven to be a highly non-trivial task [6, 18, 20, 25, 14, 17, 15]. All these works rely directly or indirectly on Sinclair's multi-commodity flow method [29]. In this approach one defines a multi-commodity flow in the state space graph of the switch chain that routes a common amount of flow between any two states of the switch chain. If this can be done in such a way that no edge of the state space graph gets *congested* too severely, then the switch chain is rapidly mixing. The analyses [6, 18, 20, 25, 14] all use the fact that the transition probabilities of the switch chain are polynomially bounded. This property does not hold for the Curveball chain, and therefore one cannot directly use the multi-commodity flows of the switch chain analyses in order to prove that the Curveball chain is rapidly mixing.[2]

Our comparison analysis is a special case of the classical comparison framework developed largely by Diaconis and Saloff-Coste and is based on so-called Dirichlet form comparisons of Markov chains, see, e.g., [7, 8], and also Quastel [27]. See also the expository paper by Dyer, Goldberg, Jerrum and Martin [11]. As the stationary distributions are uniform for all our Markov chains, we can use a more direct, but equivalent, framework based on positive semidefiniteness. We briefly elaborate on this equivalence in Appendix B.

[1] The lazy version of a reversible Markov chain with transition matrix P is the chain with transition matrix $(P + I)/2$, that is guaranteed to only have non-negative eigenvalues. This is mostly done to simplify the use of Sinclair's method [29], one of the most well-known methods for proving rapid mixing of a Markov chain.

[2] This was also briefly mentioned in [4], where the mixing time of the Curveball chain was raised as an open problem.

Finally, the transition matrix of the Curveball Markov chain is a special case of a heat-bath Markov chain under the definition of Dyer, Greenhill and Ullrich [12]. Our work partially builds on [12] in the sense that we compare a Markov chain, with a similar decomposition property as in the definition of a heat-bath chain, to its heat-bath variant.

2 Preliminaries

Let $\mathcal{M} = (\Omega, P)$ be an ergodic, time-reversible Markov chain[3] over state space Ω with transition matrix P and stationary distribution π. We write $P_x^t = P^t(x, \cdot)$ for the distribution over Ω at time step t given that the initial state is $x \in \Omega$. It is well-known that the matrix P only has real eigenvalues $1 = \lambda_0 > \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_{N-1} > -1$, where $N = |\Omega|$. Moreover, we define $\lambda_* = \max\{\lambda_1, |\lambda_{N-1}|\}$ as the second-largest eigenvalue of P. The *variation distance* at time t with initial state x is

$$\Delta_x(t) = \max_{S \subseteq \Omega} \left| P^t(x, S) - \pi(S) \right| = \frac{1}{2} \sum_{y \in \Omega} \left| P^t(x, y) - \pi(y) \right|,$$

and the mixing time $\tau(\epsilon)$ is then defined as

$$\tau(\epsilon) = \max_{x \in \Omega} \left\{ \min\{t : \Delta_x(t') \leq \epsilon \text{ for all } t' \geq t\} \right\}.$$

A Markov chain is said to be *rapidly mixing* if the mixing time can be upper bounded by a function polynomial in $\ln(|\Omega|/\epsilon)$. It is well-known, e.g., following directly from Proposition 1 [29], that

$$\frac{1}{2} \frac{\lambda_*}{1 - \lambda_*} \ln(1/2\epsilon) \;\leq\; \tau(\epsilon) \;\leq\; \frac{1}{1 - \lambda_*} \cdot (\ln(1/\pi_*) + \ln(1/\epsilon)) \tag{1}$$

where $\pi_* = \min_{x \in \Omega} \pi(x)$. This roughly implies that the mixing time is determined by the *spectral gap* $(1 - \lambda_*)$, or its inverse, the *relaxation time* $(1 - \lambda_*)^{-1}$. Finally, we let $G_\Omega = (\Omega, A)$ be the state space graph, with an arc $(a, b) \in A$ if and only if $P(a, b) > 0$ for $a, b \in \Omega$ with $a \neq b$. If P is symmetric, we define $H_\Omega = (\Omega, E)$ as the undirected counterpart of G_Ω with $\{a, b\} \in E$ if and only if $(a, b), (b, a) \in A$ with $a \neq b$.

2.1 Johnson graphs

One class of graphs that are of particular interest in this work, are the so-called Johnson graphs. For given integers $1 \leq q \leq p$, the undirected Johnson graph $J(p, q)$ contains as nodes all subsets of size q of $\{1, \ldots, p\}$, and two subsets $u, v \subseteq \{1, \ldots, p\}$ are adjacent if and only if $|u \cap v| = q - 1$. We refer the reader to [22, 3] for the following facts. The Johnson graph $J(p, q)$ is a $q(p - q)$-regular graph and the eigenvalues of its adjacency matrix are given by

$$(q - i)(p - q - i) - i \qquad \text{with multiplicity} \qquad \binom{p}{i} - \binom{p}{i - 1}$$

for $i = 0, \ldots, q$, with the convention that $\binom{p}{-1} = 0$. The following observation is included for ease of reference. It will often be used to lower bound the smallest eigenvalue of a Johnson graph.

[3] For more background on Markov chains, we refer the reader to, e.g., [24].

▶ **Proposition 1.** *Let $p, q \in \mathbb{N}$ be given. The continuous function $f : \mathbb{R} \to \mathbb{R}$ defined by*

$$f(x) = [(q-x)(p-q-x)-x] - q(p-q) = x(x-(p+1))$$

is minimized for $x^* = (p+1)/2$, with $f(x^*) = -(p+1)^2/4$.

3 Comparison framework

In this section we describe the comparison framework that will be used to compare the KTV-switch and Curveball Markov chains (Section 4), and to compare the Curveball and the parallel Curveball chain (Section 5). In general, we consider an ergodic (irreducible) Markov chain $\mathcal{M} = (\Omega, P)$ with stationary distribution π, being strictly positive for all $x \in \Omega$, that can be decomposed as[4]

$$P = \sum_{a \in \mathcal{L}} \rho(a) \sum_{R \in \mathcal{R}_a} P_R \tag{2}$$

which is given by a

i) finite index set $\mathcal{L}$, and probability distribution ρ over $\mathcal{L}$,

ii) partition $\mathcal{R}_a = \cup R_{\ell,a}$ of Ω for $a \in \mathcal{L}$,

and where the restriction of a matrix P_R to the rows and columns of $R = R_{\ell,a}$ defines the transition matrix of an ergodic, time-reversible Markov chain on R (and is zero elsewhere), with stationary distribution $\tilde{\pi}_R(x) = \pi(x)/\pi(R)$ for $x \in R$. We use $1 = \lambda_0^R \geq \lambda_1^R \geq \cdots \geq \lambda_{|R|-1}^R$ to denotes its eigenvalues. Note that these are also eigenvalues of P_R and that all other eigenvalues of P_R are zeros (as all rows and columns not corresponding to elements in R only contain zeros). Note that the chain $\mathcal{M}$ proceeds by drawing an index a from the set $\mathcal{L}$, and then performs a transition in the Markov chain on the set R that the current state is in.

The *heat-bath variant* $\mathcal{M}_{heat}$ of the chain $\mathcal{M}$ is given by the transition matrix

$$P_{heat} = \sum_{a \in \mathcal{L}} \rho(a) \sum_{R \in \mathcal{R}_a} \mathbf{1} \cdot \sigma_R \tag{3}$$

with σ_R is a row-vector given by $\sigma_R(x) = \tilde{\pi}_R(x)$ if $x \in R$ and zero otherwise, and $\mathbf{1}$ the all-ones column vector. Intuitively, the chain $\mathcal{M}_{heat}$ proceeds by drawing an index a from $\mathcal{L}$ and then drawing a state x in R with probability $\tilde{\pi}_R(x)$. It can be shown that $\mathcal{M}_{heat}$ is an ergodic Markov chain whenever $\mathcal{M}$ is ergodic, as the state space graph of $\mathcal{M}$ is a subgraph of the state space graph of $\mathcal{M}_{heat}$. It is reversible by construction [12]. The eigenvalues of P_{heat} are always non-negative as was shown in [12].

▶ **Theorem 2.** *Let $\mathcal{M} = (\Omega, P)$ be a Markov chain as in (2) with the property that $\lambda_0^R, \ldots, \lambda_{|R|-1}^R \geq 0$ for all $R \in \mathcal{R}_a$. Let $\mathcal{M}_{heat} = (\Omega, P_{heat})$ be its heat-bath variant as in (3) and let α and β be constants with $\alpha\beta > 0$. If*

$$\alpha - \beta(1 - \lambda_i^R) \geq 0, \tag{4}$$

for all $R \in \mathcal{R}_a$ and $i \in (1, \ldots, |R|-1)$, then P only has non-negative eigenvalues and

$$\frac{1}{\alpha}\frac{1}{1-\lambda_*^{heat}} \leq \frac{1}{\beta}\frac{1}{1-\lambda_*}, \tag{5}$$

where $\lambda_*^{(heat)}$ is the second largest eigenvalue of $P_{(heat)}$. For $\alpha = \beta = 1$, we find $(1-\lambda_*^{heat})^{-1} \leq (1-\lambda_*)^{-1}$.

[4] This description is almost the same as that of a heat-bath chain [12], and is introduced to illustrate the conceptual idea.

The intuition behind Theorem 2 is that in order to compare the relaxation times of a Markov chain and its heat-bath variant, it suffices to compare them locally on the sets R. Note that α and β can both be negative, so that this statement can be used both to upper bound and lower bound the relaxation time of the heat-bath variant in terms of the original relaxation time. The proof of Theorem 2 can be found in Appendix A.

4 Comparing the switch and Curveball chain

In this section we prove our main result as stated in Theorem 3 below. We first introduce some notation and terminology. We are given $n, m \in \mathbb{N}$, fixed row sums $r = (r_1, \ldots, r_m)$, column sums $c = (c_1, \ldots, c_n)$, and a set of forbidden entries $\mathcal{F} \subseteq \{1, \ldots, m\} \times \{1, \ldots, n\}$. We define $\Omega = \Omega(r, c, \mathcal{F})$ as the set of all binary $m \times n$-matrices A satisfying these row and column sums, and for which $A(a, b) = 0$ if $(a, b) \in \mathcal{F}$. The set Ω is the state space of both the switch and Curveball chain. Deciding whether or not Ω is non-empty, and computing an element from it in case it is non-empty, can be done in time polynomial[5] in m and n. That is, in case Ω is non-empty, we can efficiently compute an initial state for the switch or Curveball algorithm. The precise formulations of the transition matrices P_C and P_{KTV} of the Curveball and KTV switch chain, respectively, are given later on in this section.

▶ **Theorem 3.** *Let $r = (r_1, \ldots, r_n)$ and $c = (c_1, \ldots, c_m)$ be given marginals with $n \geq 3$, $\mathcal{F}$ a set of forbidden entries, and assume that $\Omega(r, c, \mathcal{F}) \neq \emptyset$. Let P_C and P_{KTV} be the transition matrices of respectively the Curveball and KTV-switch Markov chains. Then*

$$\frac{2}{n(n-1)} \cdot (1 - \lambda_*^{KTV})^{-1} \;\leq\; (1 - \lambda_*^{C})^{-1} \;\leq\; \min\left\{1, \frac{(2r_{\max}+1)^2}{2n(n-1)}\right\} \cdot (1 - \lambda_*^{KTV})^{-1},$$

where $\lambda_*^{(KTV,C)} = \lambda_1^{(KTV,C)}$ is the second largest eigenvalue of $P_{(KTV,C)}$.

In order to prove Theorem 3, we give a novel decomposition of the state space of the KTV-switch chain. We then show that the Curveball chain is its heat-bath variant. In fact, we introduce a general γ-switch chain, as there are multiple switch-based chains in the literature, and show that the Curveball chain is the heat-bath variant of this general switch chain. The KTV-switch chain corresponds to a specific choice of γ. In the full version [5] we compare the Curveball chain with another switch-based chain for a different value of γ.

4.1 Switch chain

For a given initial binary matrix A, in every step of the Kannan-Tetali-Vempala (KTV) switch algorithm we choose two distinct rows and two distinct columns uniformly at random. If the 2×2 submatrix corresponding to these rows and columns is a *checkerboard* C_i for $i = 1, 2$, where,

$$C_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \text{and} \quad C_2 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix},$$

then the 2×2 submatrix is replaced by C_{i+1} for i modulo 2 provided the zero entries are not forbidden. That is, if the checkerboard is C_1, it is replaced by C_2, and vice versa. If the

[5] Deciding non-emptiness of Ω can be reduced to deciding if a certain auxiliary graph contains a perfect matching [31]. The latter can be done using the well-known blossom shrinking algorithm [13]. This is also mentioned in [14].

submatrix does not correspond to a checkerboard, nothing is changed. Such an operation is called a *switch*.

Matrices $A, B \in \Omega$ are *switch-adjacent for row i and j* if $A = B$ or if $A - B$ contains exactly four non-zero elements that occur on rows i and j and columns k and ℓ. Two matrices are switch-adjacent if they are switch-adjacent for some rows i and j. In the KTV-switch chain, the probability of transitioning between switch-adjacent matrices is the probability of selecting rows i and j and columns k and ℓ, and always equals $\binom{m}{2}^{-1}\binom{n}{2}^{-1}$.

We need the following additional definitions to introduce the general γ-switch chain. For $A \in \Omega$, we let A_{ij} be the $2 \times n$-submatrix formed by rows i and j, for $1 \leq i < j \leq m$. We define

$$U_{ij}(A) = \{k \in \{1, \dots, n\} : A(i,k) = 1,\ A(j,k) = 0 \text{ and } (j,k) \notin \mathcal{F}\}, \tag{6}$$

with $u_{ij}(A) = |U_{ij}(A)|$, and similarly

$$L_{ij}(A) = \{k \in \{1, \dots, n\} : A(i,k) = 0,\ A(j,k) = 1 \text{ and } (i,k) \notin \mathcal{F}\}, \tag{7}$$

with $l_{ij}(A) = |L_{ij}(A)|$. Note that $L_{ij} \cup U_{ij}$ are precisely the columns k for which A_{ij} has different values on its rows and for which (i,k) and (j,k) are both not forbidden. We will often write u_{ij} and l_{ij} instead of $u_{ij}(A)$ and $l_{ij}(A)$ for brevity.

▶ **Definition 4** (γ-switch chain)**.** Let γ be such that

$$1 - u_{ij}(A)l_{ij}(A) \cdot \gamma > 0 \tag{8}$$

for all $A \in \Omega = \Omega(r, c, \mathcal{F})$ and $1 \leq i < j \leq m$. The transition matrix of the γ-switch chain on state space Ω is given by

$$P_\gamma(A,B) = \begin{cases} \binom{m}{2}^{-1} \cdot \gamma & \text{if } A \neq B \text{ are switch-adjacent,} \\ \binom{m}{2}^{-1} \sum_{1 \leq i < j \leq m} (1 - u_{ij}(A)l_{ij}(A) \cdot \gamma) & \text{if } A = B, \\ 0 & \text{otherwise.} \end{cases}$$

Note that the transition probability for switch-adjacent matrices is the same everywhere in the state space, and does not depend on the matrices A and B. In particular, the transition matrix P_γ is symmetric and hence the chain is reversible with respect to the uniform distribution. The factor $2/(m(m-1))$ is included for notational convenience. The γ-switch chain can roughly be interpreted as follows. We first choose two distinct rows i and j uniformly at random, and then transition to a different matrix switch-adjacent for rows i and j, of which there are $u_{ij}l_{ij}$ possibilities and where every matrix has probability γ of being chosen; with probability $1 - u_{ij}l_{ij}\gamma$ we do nothing. Taking $\gamma = 2/(n(n-1))$ we obtain the KTV-switch chain [23].

▶ Remark. We always assume the γ-switch chain is irreducible for given r, c and $\mathcal{F}$ (it is clearly always aperiodic, symmetric and finite). Irreducibility is for instance guaranteed in case there are no forbidden entries [28]; or in case $n = m \geq 4$, with $\mathcal{F}$ is the set of diagonal entries and regular marginals $c_i = r_i = d$ for some given $d \geq 1$ [18]. A characterization for irreducibility in the case where $\mathcal{F}$ is the set of diagonal entries is given in [1]. Note that the condition of irreducibility is independent of the value of γ.

4.2 Curveball chain

The Curveball chain proceeds as follows. In every step two rows are chosen uniformly at random from A as in the γ-switch algorithm. Then, a so-called *binomial trade* is performed. In such a trade, we first find all the columns $U_{ij}(A)$ and $L_{ij}(A)$. For example, if $n = 6$ and the $2 \times n$ submatrix is given by

$$\begin{pmatrix} 1 & \mathbf{1} & 0 & \mathbf{0} & \mathbf{0} & \mathbf{1} \\ 1 & \mathbf{0} & 0 & \mathbf{1} & \mathbf{1} & \mathbf{0} \end{pmatrix},$$

then we consider the (auxiliary) submatrix

$$\begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

given by the second, fourth, fifth and sixth column. We now uniformly at random draw a $2 \times (u_{ij} + l_{ij})$ matrix with columns sums equal to 1, and row sums equal to u_{ij} and l_{ij}. Note that there are $\binom{u_{ij}+l_{ij}}{u_{ij}}$ possible choices, hence the name binomial trade. We then replace the (auxiliary) submatrix with this new submatrix in A. Note that such a drawing can be obtained by uniformly choosing u_{ij} out of $u_{ij} + l_{ij}$ column indices.

Similarly as in the switch chain definition, two matrices A and B are called *trade-adjacent for rows i and j* if $A = B$ or if B can be obtained from A using one binomial trade operation on rows i and j. Two matrices are trade-adjacent if they are trade-adjacent for some row pair.

The Curveball chain $\mathcal{M}_C$ then equals (Ω, P_C) with $\Omega = \Omega(r, c, \mathcal{F})$ and

$$P_C(A, B) = \begin{cases} \binom{m}{2}^{-1} \cdot \binom{u_{ij}+l_{ij}}{u_{ij}}^{-1} & \text{if } A \neq B \text{ are trade-adjacent,} \\ \binom{m}{2}^{-1} \sum_{1 \leq i < j \leq m} \binom{u_{ij}+l_{ij}}{u_{ij}}^{-1} & \text{if } A = B, \\ 0 & \text{otherwise.} \end{cases}$$

4.3 State space decomposition

We next explain how the switch and Curveball chain fit in the comparison framework. The index set $\mathcal{L} = \{(i, j) : 1 \leq i < j \leq m\}$ is the set of all pairs of distinct rows, and ρ is the uniform distribution over $\mathcal{L}$, that is, $\rho(i, j) = \binom{m}{2}^{-1}$ for all $(i, j) \in \mathcal{L}$. The partitions $\mathcal{R}_{(i,j)}$ for $(i, j) \in \mathcal{L}$ are based on the notion of a binomial neighborhood, as defined in [32].

▶ **Definition 5** (Binomial neighborhood)**.** For a fixed binary matrix A and row-pair (i, j), the (i, j)-binomial neighborhood $\mathcal{N}_{ij}(A)$ of A is the set of matrices that can be reached by only applying switches on rows i and j. That is, $\mathcal{N}_{ij}(A)$ contains all matrices that are trade-adjacent to A for rows i and j.

Note that for $B \in \mathcal{N}_{ij}(A)$ we have $U_{ij}(A) \cup L_{ij}(A) = U_{ij}(B) \cup L_{ij}(B)$ and furthermore $u_{ij}(A) = u_{ij}(B)$ and $l_{ij}(A) = l_{ij}(B)$.

Next we will discuss the structure and properties of these binomial neighborhoods. This discussion will culminate into Theorem 7 describing the switch and Curveball chains as being of the forms (2) and (3). Note that we have $A \in \mathcal{N}_{ij}(A)$; if $B \in \mathcal{N}_{ij}(A)$, then $A \in \mathcal{N}_{ij}(B)$ [32]; and, if $A \in \mathcal{N}_{ij}(B)$, $B \in \mathcal{N}_{ij}(C)$, then $A \in \mathcal{N}_{ij}(C)$. That is, the relation $\sim_{ij}$ defined by $a \sim_{ij} b$ if and only if $a \in \mathcal{N}_{ij}(b)$, is an equivalence relation on Ω. The equivalence classes of $\sim_{ij}$ define the sets $\mathcal{R}_{(i,j)}$.

Furthermore, two matrices $A, B \in \Omega$ can be part of *at most* one common binomial neighborhood. This follows directly from the observation that if $B \in \mathcal{N}_{ij}(A) \backslash \{A\}$, then A and

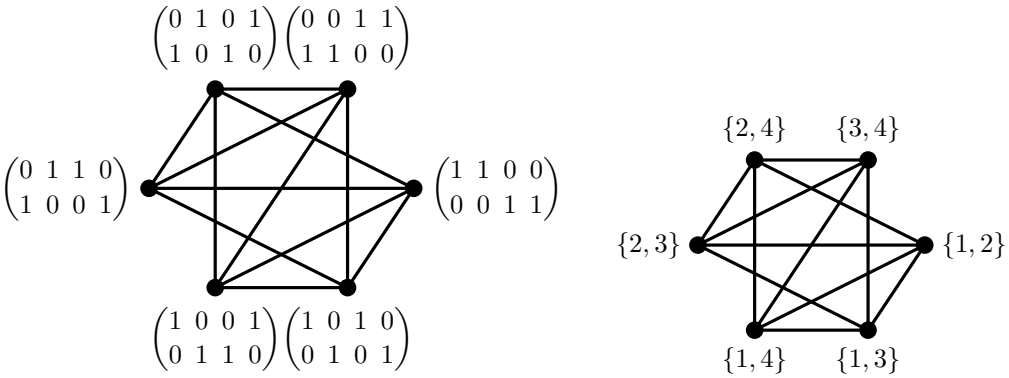


Figure 1 The induced subgraph H for the switch chain on the $(1,2)$-binomial neighborhood of A. On the left we have indexed the nodes by the submatrices of the first four columns, and on the right by label sets, indicating the positions of the 1's on the top row (i.e., row 1).

B differ on precisely rows i and j, so switches using any other pair of rows $\{k,\ell\} \neq \{i,j\}$ can never transform A into B, see [32]. Finally, since $u_{ij}(A) = u_{ij}(B)$ and $l_{ij}(A) = l_{ij}(B)$ when A and B are part of the same binomial neighborhood, these numbers are only neighborhood-dependent, and not element-dependent within a fixed neighborhood. Observe that

$$|\mathcal{N}| = \binom{u_{ij}+l_{ij}}{u_{ij}}.$$

A crucial observation now is that the undirected state space graph H of the γ-switch chain induced on a binomial neighborhood $\mathcal{N}_{ij}$ is isomorphic to a Johnson graph $J(u_{ij}+l_{ij}, u_{ij})$ whenever $u_{ij}, l_{ij} \geq 1$ (see Section 2.1 for notation and definition).[6] To see this, note that every element in the binomial neighborhood $\mathcal{N}_{ij}(A)$ can be represented by the set of indices $U_{ij}(A)$. The set $\{1,\ldots,l_{ij}+u_{ij}\}$ here is then the set of indices of $U_{ij}(A) \cup L_{ij}(A)$. Indeed, matrices $A \neq B$ are switch-adjacent for rows i and j if $U_{ij}(A) \cap U_{ij}(B) = u_{ij} - 1$.

▶ **Example 6.** Consider the binary matrix

$$A = \begin{pmatrix}0&1&1&0&1&0&1\\1&0&0&1&1&0&1\\0&1&0&0&0&1&1\end{pmatrix}$$

and the 2×7-submatrix formed by rows 1 and 2, which is

$$A_{12} = \begin{pmatrix}0&1&1&0&1&0&1\\1&0&0&1&1&0&1\end{pmatrix}.$$

For sake of simplicity, we (uniquely) describe every element of the $(1,2)$-binomial neighborhood $\mathcal{N}_{12}(A)$ by the first four columns (precisely those with column sums equal to one in the submatrix). For the switch chain, the induced subgraph of the undirected state space graph H on the $(1,2)$-binomial neighborhood of A, the Johnson graph $J(4,2)$ is given in Figure 1.

▶ Remark. A fixed binomial neighborhood is reminiscent of the Bernoulli-Laplace Diffusion model, see, e.g., [9, 10] for an analysis of this model. In this model there are two bins

[6] If either $u_{ij} = 0$ or $l_{ij} = 0$ it consists of a single binary matrix.

with respectively k and $n-k$ balls, and in every transition two randomly chosen balls, one from each bin, are interchanged between the bins. Indeed, the state space graph is then a Johnson graph [10]. The transition probabilities are different, due to the non-zero holding probabilities in the switch algorithm, but the eigenvalues of this Markov chain are related to the eigenvalues of the switch Markov chain on a fixed binomial neighborhood, see also [9, 10].

Informally, the Markov chain resulting from always deterministically choosing rows i and j in the switch algorithm, is the disjoint union of smaller Markov chains each with a state space graph isomorphic to some Johnson graph. For a binomial neighborhood $\mathcal{N} = \mathcal{N}_{ij}(A)$ for given $i < j$ and $A \in \Omega$, the undirected graph $H_{\mathcal{N}} = (\Omega, E_{\mathcal{N}})$ is the graph where $E_{\mathcal{N}}$ forms the edge-set of the Johnson graph $J(u_{ij} + l_{ij}, u_{ij})$ on $\mathcal{N} \subseteq \Omega$, and where all binary matrices $B \in \Omega \setminus \mathcal{N}$ are isolated nodes. We use $M(H_{\mathcal{N}})$ to denote its adjacency matrix. The discussion above leads to the following theorem, where we define I_S as the identity matrix on S, and we define J_S as the all-ones matrix on S, that is $J_S(x, y) = 1$ if $x, y \in S$ and zero elsewhere.

▶ **Theorem 7.** *The transition matrix P_γ of the γ-switch chain is of the form (2) namely*

$$P_\gamma = \sum_{1 \le i < j \le m} \binom{m}{2}^{-1} \sum_{\mathcal{N} \in \mathcal{R}_{(i,j)}} \left((1 - u_{ij} l_{ij} \cdot \gamma) \cdot I_{\mathcal{N}} + \gamma \cdot M(H_{\mathcal{N}})\right). \tag{9}$$

The heat-bath variant of the γ-switch chain is given by the Curveball chain, and can be written as

$$P_C = \sum_{1 \le i < j \le m} \binom{m}{2}^{-1} \sum_{\mathcal{N} \in \mathcal{R}_{(i,j)}} \binom{u_{ij} + l_{ij}}{u_{ij}}^{-1} J_{\mathcal{N}}. \tag{10}$$

Proof. The decomposition in (9) follows from the discussion above, and (8) guarantees that the matrix $(1 - u_{ij} l_{ij} \cdot \gamma) \cdot I_{\mathcal{N}} + \gamma \cdot M(H_{\mathcal{N}})$ indeed defines the transition matrix of a Markov chain for every $\mathcal{N}$. Moreover, remember that the γ-switch chain has uniform stationary distribution π over Ω. Indeed, for a binomial neighborhood $\mathcal{N} = \mathcal{N}_{ij}(A)$ for given $i < j$ and $A \in \Omega$, the vector $\sigma_{\mathcal{N}}$ as used in (3) is then given by

$$\sigma_{\mathcal{N}}(x) = \frac{\pi(x)}{\pi(\mathcal{N})} = \frac{1}{|\Omega|} \cdot \frac{|\Omega|}{|\mathcal{N}|} = \frac{1}{|\mathcal{N}|} = \binom{u_{ij} + l_{ij}}{u_{ij}}^{-1}$$

if $x \in \mathcal{N}$, and zero otherwise. This implies that $\mathbf{1} \cdot \sigma_{\mathcal{N}} = \binom{u_{ij}+l_{ij}}{u_{ij}}^{-1} J_{\mathcal{N}}$ as desired. ◀

As a by-product of this decomposition, we now show that the KTV-switch chain [23] only has non-negative eigenvalues when $n \ge 3$. This is of independent interest, as discussed in the introduction.

▶ **Theorem 8.** *The transition matrix of the KTV-switch Markov chain only has non-negative eigenvalues when $n \ge 3$.*

Proof. The KTV-switch chain is exactly the γ-switch chain with $\gamma = 2/(n(n-1))$. As the product $u_{ij}(A) l_{ij}(A)$ can be at most $n^2/4$ for any $A \in \Omega$ and $1 \le i < j \le m$, we see that γ satisfies (8) when $n \ge 3$. To show that P_{KTV} has all non-negative eigenvalues we show that the property assumed in Theorem 2 is satisfied by showing that the matrices

$$Y_{\mathcal{N}} = \left[1 - u_{ij} l_{ij} \cdot \binom{n}{2}^{-1}\right] I_{\mathcal{N}} + \binom{n}{2}^{-1} M(H_{\mathcal{N}})$$

have all non-negative eigenvalues. Theorem 2 then implies that P_{KTV} also only has non-negative eigenvalues. For any eigenvalue λ of this submatrix, we have $\lambda = 1+(\mu-u_{ij}l_{ij})\binom{n}{2}^{-1}$ where $\mu = \mu(\lambda)$ is an eigenvalue of the Johnson graph $J(u_{ij}+l_{ij}, u_{ij})$ on $\mathcal{N}$. In particular, using Proposition 1 with $p = u_{ij}+l_{ij}$ and $q = u_{ij}$, we get $(\mu - u_{ij}l_{ij}) \geq -\frac{1}{4}(u_{ij}+l_{ij}+1)^2 \geq -\frac{1}{4}(n+1)^2$ using that $0 \leq u_{ij}+l_{ij} \leq n$. Therefore, when $n \geq 5$, we have $\lambda \geq 1-(n+1)^2/(2n(n-1)) \geq 0$. The cases $n = 3, 4$ can be checked with some elementary arguments. This is left to the reader. ◀

We conclude this section with the proof of Theorem 3.

Proof of Theorem 3. Let $\mathcal{N} = \mathcal{N}_{ij}(A)$ for given $i < j$ and $A \in \Omega$. Note that the upper bound $(1-\lambda_*^{KTV})^{-1}$ follows from Theorem 2 with $\alpha = \beta = 1$ for which (5) holds as was shown in Theorem 8. We apply Theorem 2 for two pairs (α, β) to obtain the remaining upper and lower bound.

Case 1: $\alpha = 1$ and $\beta = (2n(n-1))/((2r_{\max}+1)^2)$. We show that condition (4) is satisfied. That is, we show that

$$\lambda = 1-\beta\left(1-\left(1+(\mu-u_{ij}\cdot l_{ij})\binom{n}{2}^{-1}\right)\right) = 1+\beta(\mu-u_{ij}\cdot l_{ij})\binom{n}{2}^{-1} \geq 0$$

for any $\mu = \mu(\lambda)$ that is an eigenvalue of the Johnson graph $J(u_{ij}+l_{ij}, u_{ij})$. Again, using Proposition 1 in order to lower bound the quantity $(\mu - u_{ij}\cdot l_{ij})$, we find

$$1+\beta\cdot(\mu-u_{ij}\cdot l_{ij})\binom{n}{2}^{-1} \geq 1-\frac{\beta}{4}(u_{ij}+l_{ij}+1)^2\binom{n}{2}^{-1} \geq 1-\frac{\beta}{4}(2r_{\max}+1)^2\binom{n}{2}^{-1} \geq 0,$$

using the fact that $0 \leq u_{ij}+l_{ij} \leq 2r_{\max}$ and the choice of β. Hence we find the second part of the upper bound.

Case 2: $\alpha = -1$ and $\beta = -(n(n-1))/2$. We have to show that

$$\lambda = \binom{n}{2}\left(1-\left(1+(\mu-u_{ij}\cdot l_{ij})\binom{n}{2}^{-1}\right)\right)-1 = u_{ij}\cdot l_{ij}-\mu-1 \geq 0$$

for all $\mu = \mu(k) = (u-k)(\ell-k)-k$ where $k = 1, \ldots, u$. Note that the eigenvalue $u_{ij}\cdot l_{ij}$ for the case $k = 0$ yields the largest eigenvalue $1 = \lambda_0^{\mathcal{N}}$ of $Y_{\mathcal{N}}$, and does not have to be considered here. The maximum over $k = 1, \ldots, u$ is then attained for $k = 1$, and we have $u_{ij}\cdot l_{ij}-\mu-1 \geq u_{ij}\cdot l_{ij}-((u_{ij}-1)(l_{ij}-1)-1)-1 = u_{ij}+l_{ij}-1 \geq 0$, since $u_{ij}, l_{ij} \geq 1$. This gives us the lower bound and finishes the proof. ◀

5 Parallelism in the Curveball chain

In this section we discuss an additional application of the comparison framework in Section 3. As a binary matrix is only adjusted on two rows at the time in the Curveball algorithm, one might perform multiple binomial trades in parallel on distinct pairs of rows [4]. To be precise, in every step of the so-called *k-Curveball algorithm*, we choose a set of $k \leq \lfloor m/2 \rfloor$ disjoint pairs of rows uniformly at random and perform a binomial trade on every pair (see Section 4.2). For $k = \lfloor m/2 \rfloor$ this corresponds to the Global Curveball algorithm described in [4]. We show that the *k-Curveball chain*, resulting from the k-Curveball algorithm, is the heat-bath variant of the Curveball chain. We use the notation as given in Section 3.

The index set $\mathcal{L} = \mathcal{L}(k)$ is the collection of all sets containing k pairwise disjoint sets of two rows, i.e.,

$$\Big\{\{(1_c, 1_d), (2_c, 2_d), \ldots, (k_c, k_d)\} \ : \ 1_c, 1_d, \ldots, k_c, k_d \in [m],\ |\{1_c, 1_d, 2_c, 2_d, \ldots, k_c, k_d\}| = 2k\Big\},$$

and ρ is the uniform distribution over $\mathcal{L}$. For a fixed collection $\kappa \in \mathcal{L}(k)$, we define the κ-neighborhood $\mathcal{N}_\kappa(A)$ of a binary matrix $A \in \Omega$ as the set of binary matrices $B \in \Omega$ that can be obtained from A by binomial trade-operations only involving the row-pairs in κ. Formally speaking, we have $B \in \mathcal{N}_\kappa(A)$ if and only if there exist binary matrices A_ℓ for $\ell = 0, \ldots, k-1$, so that

$$A_{\ell+1} \in \mathcal{N}_{(\ell+1)_c,(\ell+1)_d}(A_\ell)$$

where $A = A_0$ and $B = A_k$. Note that the matrices A_ℓ might not all be pairwise distinct, as A and B could already coincide on certain pairs of rows in κ. Also note that $u_{i_c i_d}(A) = u_{i_c i_d}(B)$ and $l_{i_c i_d}(A) = l_{i_c i_d}(B)$ if $B \in \mathcal{N}_\kappa(A)$ for $i = 1, \ldots, k$. It is not hard to see that such a neighborhood is isomorphic to a Cartesian product $W_1 \times W_2 \times \cdots \times W_k$ of finite sets[7] $W_1, \ldots, W_k$ with

$$|W_i| = \binom{u_{i_c i_d} + l_{i_c i_d}}{u_{i_c i_d}}.$$

Moreover, the relation $\sim_\kappa$ defined by $a \sim_\kappa b$ if and only if $b \in \mathcal{N}_\kappa(a)$ defines an equivalence relation, and its equivalence classes give the set $\mathcal{R}_\kappa$. We now consider the following artificial formulation of the original Curveball chain: we first select k pairs of distinct rows uniformly at random, and then we choose one of those pairs uniformly at random and apply a binomial trade on that pair. It should be clear that this generates the same Markov chain as when we directly select a pair of distinct rows uniformly at random. For $\mathcal{N}_\kappa \in \mathcal{R}_\kappa$ the matrix $P_{\mathcal{N}_\kappa}$ restricted to the rows and columns in $\mathcal{N}_\kappa$ is then the transition matrix of a Markov chain over $W_1 \times \cdots \times W_k$, where in every step we choose an index $i \in [k]$ uniformly at random and make a transition in W_i based on the (uniform) transition matrix

$$Q_i = \binom{u_{i_c i_d} + l_{i_c i_d}}{u_{i_c i_d}}^{-1} J$$

where J is the all-ones matrix of appropriate size. More formally, the matrix $P_{\mathcal{N}_\kappa}$ restricted to the columns and rows in $\mathcal{N}_\kappa$ is given by

$$\frac{\sum_{i=1}^{k} \left[\otimes_{j=1}^{i-1} \mathcal{I}_j\right] \otimes Q_i \otimes \left[\otimes_{j=i+1}^{k} \mathcal{I}_j\right]}{k}, \tag{11}$$

forming a transition matrix on $\mathcal{N}_\kappa$, and is zero elsewhere. Here $\mathcal{I}_j$ is the identity matrix with the same size as Q_j and $\otimes$ the usual tensor product. The eigenvalues of the matrix in (11) are given by

$$\lambda_{\mathcal{N}_\kappa} = \left\{ \frac{1}{k} \sum_{i=1}^{k} \lambda_{j_i,i} : 0 \le j_i \le |W_i| - 1 \right\} \tag{12}$$

where $1 = \lambda_{0,i} \ge \lambda_{1,i} \ge \cdots \ge \lambda_{|W_i|-1,i}$ are the eigenvalues[8] of Q_i for $i = 1, \ldots, k$. It then follows that

$$P_C = \sum_{\kappa \in \mathcal{L}(k)} \frac{1}{|\mathcal{L}(k)|} \sum_{\mathcal{N}_\kappa \in \mathcal{R}_\kappa} P_{\mathcal{N}_\kappa}$$

---

[7] That is, the elements of W_i describe a matrix on row-pair (i_c, i_d).

[8] See, e.g., [16] for a similar argument regarding the transition matrix, and eigenvalues, of a Markov chain of this form. These statements follow directly from elementary arguments involving tensor products.

which is of the form (2). For $k = 1$, we get back the description of the previous section. Now, its heat-bath variant is precisely the k-Curveball Markov chain

$$P_{k,C} = \sum_{\kappa \in \mathcal{L}(k)} \frac{1}{|\mathcal{L}(k)|} \sum_{\mathcal{N}_\kappa \in \mathcal{R}_\kappa} \frac{1}{|\mathcal{N}_\kappa|} J_{\mathcal{N}_\kappa},$$

where

$$|\mathcal{N}_\kappa| = \prod_{i=1}^{k} \binom{u_{i_c i_d} + l_{i_c i_d}}{u_{i_c i_d}}^{-1}$$

as, roughly speaking, for a fixed neighborhood $\mathcal{N}_\kappa$, the k-Curveball chain is precisely the uniform sampler over such a neighborhood.

▶ **Theorem 9.** *We have* $(1 - \lambda_*^C)^{-1}/k \leq (1 - \lambda_*^{k,C})^{-1} \leq (1 - \lambda_*^C)^{-1}$ *where* $\lambda_*^{k,C}$ *is the second-largest eigenvalue of the* k*-Curveball chain, and* λ_*^C *the second-largest eigenvalue of the original (*1*-)Curveball chain.*

Proof. The upper bound follows from Theorem 2, with $\alpha = \beta = 1$, as the eigenvalues of all the Q_i are non-negative, and therefore (12) implies that the eigenvalues of the matrix in (11) are also non-negative. For the lower bound, we take $\alpha = -1$ and $\beta = -k$. That is, we have to show that $-1 + k(1 - \mu) \geq 0$ with $\mu \in \lambda_{\mathcal{N}_\kappa} \setminus \{1\}$ as in (12). It is not hard to see that the second-largest eigenvalue in $\lambda_{\mathcal{N}_\kappa}$ is $(k - 1)/k$, as the eigenvalues of every fixed Q_i are $1 = \lambda_{0,i} > \lambda_{1,i} = \cdots = \lambda_{|W_i|-1} = 0$. This implies that $-1+k(1-\mu) \geq -1+k(1-(k-1)/k) = 0$ for all $\mu \in \lambda_{\mathcal{N}_\kappa} \setminus \{1\}$. ◀

In general, the upper bound is tight for certain (degenerate) cases, that is, parallelism in the Curveball chain does not necessarily guarantee an improvement in its relaxation time. E.g., take column marginals $c_i = 1$ for $i = 1, \ldots, n$, and row-marginals $r_1 = r_2 = n/2$ and $r_3 = r_4 = 0$, and consider $k = 2$.

6 Discussion

We believe similar ideas as in this work can be used to prove that the Curveball chain is rapidly mixing for the sampling of undirected graphs with given degree sequences [4], whenever one of the switch chains is rapidly mixing for those degree sequences. We leave this for future work, as the proof we have in mind is a bit more involved, but of a very similar nature as the ideas described here.

It should be noted that the main conclusion of our work is not that the Curveball algorithm is necessarily better than the switch-based approaches. In particular, the improvement in relaxation time in Theorem 3, when the maximum row sum is small compared to n, is mostly caused by the fact that the KTV-switch chain is a bad choice of implementation here (as the holding probability of a state in the Markov chain is relatively large in this case). There exist other implementations of the switch chain that are more efficient than the KTV switch chain for certain marginals. For example, the so-called *edge-switch* chain [6, 20]. Here, instead of choosing two rows and columns uniformly at random, one chooses two one-entries of a binary matrix uniformly at random (in order to reduce the probability of staying in the same state of the Markov chain). We give a comparison between these chains in the full version [5]. An interesting direction for future work is to give a better comparison than that in [5]. Although we believe the Curveball chain will outperform any switch-based chain for certain marginals, it not obvious for which marginals this is true. For example, it is not clear to us if this is true

in the case of sampling regular directed graphs with in- and out-degree some small constant. However, for graphs with large regular degrees we expect the Curveball chain to be better.

Moreover, one step of the Curveball algorithm is computationally more expensive than one step of a switch-based algorithm, so although the relaxation time of the Curveball chain might be better than a switch-based chain, this does not automatically imply that the overall running time of the Curveball algorithm is better than that of a switch-based algorithm. Nevertheless, we believe that our results are a first theoretical step for speeding up switch-based Markov chains for sampling bipartite graphs with a given degree sequence.

References

1 Annabell Berger and Matthias Müller-Hannemann. Uniform sampling of digraphs with a fixed degree sequence. In *Graph Theoretic Concepts in Computer Science: 36th International Workshop, WG 2010, Zarós, Crete, Greece, June 28-30, 2010 Revised Papers*, pages 220–231, 2010.

2 Ivona Bezáková, Nayantara Bhatnagar, and Dana Randall. On the Diaconis-Gangolli Markov chain for sampling contingency tables with cell-bounded entries. In *Computing and Combinatorics: 15th Annual International Conference, COCOON 2009 Niagara Falls, NY, USA, July 13-15, 2009 Proceedings*, pages 307–316, 2009.

3 Andries E. Brouwer and Willem H. Haemers. *Spectra of graphs*. Universitext. Springer New York, 2011.

4 Corrie Jacobien Carstens, Annabell Berger, and Giovanni Strona. Curveball: a new generation of sampling algorithms for graphs with fixed degree sequence. *CoRR*, abs/1609.05137, 2016. URL: `http://arxiv.org/abs/1609.05137`.

5 Corrie Jacobien Carstens and Pieter Kleer. Comparing the switch and curveball Markov chains for sampling binary matrices with fixed marginals. *CoRR*, abs/1709.07290, 2017. `arXiv:1709.07290`.

6 Colin Cooper, Martin E. Dyer, and Catherine S. Greenhill. Sampling regular graphs and a peer-to-peer network. *Combinatorics, Probability & Computing*, 16(4):557–593, 2007. `doi:10.1017/S0963548306007978`.

7 Persi Diaconis and Laurent Saloff-Coste. Comparison techniques for random walk on finite groups. *The Annals of Probability*, 21(4):2131–2156, 10 1993.

8 Persi Diaconis and Laurent Saloff-Coste. Comparison theorems for reversible Markov chains. *The Annals of Applied Probability*, 3(3):696–730, 08 1993. `doi:10.1214/aoap/1177005359`.

9 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.

10 Peter Donnelly, Peter Lloyd, and Aidan Sudbury. Approach to stationarity of the Bernoulli-Laplace diffusion model. *Advances in Applied Probability*, 26(3):715–727, 1994.

11 Martin Dyer, Leslie Ann Goldberg, Mark Jerrum, and Russell Martin. Markov chain comparison. *Probab. Surveys*, 3:89–111, 2006.

12 Martin Dyer, Catherine Greenhill, and Mario Ullrich. Structure and eigenvalues of heat-bath Markov chains. *Linear Algebra and its Applications*, 454:57–71, 2014.

13 Jack Edmonds. Paths, trees, and flowers. *Canadian journal of mathematics*, 17(3):449–467, 1965.

14 Péter L. Erdős, Sándor Z. Kiss, István Miklós, and Lajos Soukup. Approximate counting of graphical realizations. *PLOS ONE*, 10(7):1–20, 2015.

15 Péter L. Erdős, Tamás Róbert Mezei, and István Miklós. Efficiently sampling the realizations of irregular, but linearly bounded bipartite and directed degree sequences. *CoRR*, abs/1712.01709, 2017. URL: `https://arxiv.org/abs/1712.01709`.

16 Péter L Erdős, István Miklós, and Zoltán Toroczkai. A decomposition based proof for fast mixing of a Markov chain over balanced realizations of a joint degree matrix. *SIAM Journal on Discrete Mathematics*, 29(1):481–499, 2015.

17 Péter L. Erdős, István Miklós, and Zoltán Toroczkai. New classes of degree sequences with fast mixing swap Markov chain sampling. *CoRR*, abs/1601.08224, 2016. URL: `http://arxiv.org/abs/1601.08224`.

18 Catherine S. Greenhill. A polynomial bound on the mixing time of a Markov chain for sampling regular directed graphs. *Electronic Journal of Combinatorics*, 18(1), 2011.

19 Catherine S. Greenhill. Making Markov chains less lazy. *CoRR*, abs/1203.6668, 2013. URL: `https://arxiv.org/abs/1203.6668`.

20 Catherine S. Greenhill. The switch Markov chain for sampling irregular graphs: extended abstract. In *Proceedings of the Twenty-sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1564–1572, 2015.

21 Catherine S. Greenhill and Matteo Sfragara. The switch Markov chain for sampling irregular graphs and digraphs. *Theoretical Computer Science*, 719:1–20, 2018.

22 Derek A. Holton and John Sheehan. *The Petersen graph*. Cambridge University Press Cambridge, 1993.

23 Ravi Kannan, Prasad Tetali, and Santosh Vempala. Simple Markov-chain algorithms for generating bipartite graphs and tournaments. *Random Structures and Algorithms*, 14(4):293–308, 1999.

24 David A. Levin, Yuval Peres, and Elizabeth L. Wilmer. *Markov chains and mixing times*. American Mathematical Society, 2009.

25 István Miklós, Péter L. Erdős, and Lajos Soukup. Towards random uniform sampling of bipartite graphs with given degree sequence. *Electronic Journal of Combinatorics*, 20(1), 2013.

26 R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: simple building blocks of complex networks. *Science*, 298:824–827, 2002. `doi:10.1126/science.298.5594.824`.

27 Jeremy Quastel. Diffusion of color in the simple exclusion process. *Communications on Pure and Applied Mathematics*, 45(6):623–679, 1992. `doi:10.1002/cpa.3160450602`.

28 A. Ramachandra Rao, Rabindranath Jana, and Suraj Bandyopadhyay. A Markov chain Monte Carlo method for generating random (0, 1)-matrices with given marginals. *Sankhyā: The Indian Journal of Statistics, Series A*, 58(2):225–242, 1996. `doi:10.2307/25051102`.

29 Alistair Sinclair. Improved bounds for mixing rates of Markov chains and multicommodity flow. *Combinatorics, Probability and Computing*, 1:351–370, 1992.

30 G. Strona, D. Nappo, F. Boccacci, S. Fattorini, and J. San-Miguel-Ayanz. A fast and unbiased procedure to randomize ecological binary matrices with fixed row and column totals. *Nature Communications*, 5(4114), 2014.

31 William T. Tutte. The factors of graphs. *Canadian Journal of Mathematics*, 4(3):314–328, 1952.

32 Norman D. Verhelst. An efficient MCMC algorithm to sample binary matrices with fixed marginals. *Psychometrika*, 73(4):705, 2008.

33 Fuzhen Zhang. *Matrix theory: basic results and techniques*. Universitext. Springer Berlin, 1999.

A Missing proofs

Theorem 2. *Let $\mathcal{M} = (\Omega, P)$ be a Markov chain as in (2) with the property that $\lambda_0^R, \ldots, \lambda_{|R|-1}^R \geq 0$ for all $R \in \mathcal{R}_a$. Let $\mathcal{M}_{heat} = (\Omega, P_{heat})$ be its heat-bath variant as in (3) and let α and β be constants with $\alpha\beta > 0$. If*

$$\alpha - \beta(1 - \lambda_i^R) \geq 0, \tag{4}$$

for all $R \in \mathcal{R}_a$ and $i \in (1, \ldots, |R| - 1)$, then P only has non-negative eigenvalues and

$$\frac{1}{\alpha}\frac{1}{1 - \lambda_*^{heat}} \leq \frac{1}{\beta}\frac{1}{1 - \lambda_*}, \tag{5}$$

where $\lambda_*^{(heat)}$ is the second largest eigenvalue of $P_{(heat)}$. In particular, for $\alpha = \beta = 1$, we find $(1 - \lambda_*^{heat})^{-1} \leq (1 - \lambda_*)^{-1}$.

We will use Propositions 10 and 11 in the proof of Theorem 2. Our proof makes use of positive semi-definite matrices; a symmetric real-valued matrix A is positive semidefinite if all its eigenvalues are non-negative, this is denoted by $A \succeq 0$.

▶ **Proposition 10** ([33]). *Let X, Y be symmetric $\ell \times \ell$ matrices. If $X - Y \succeq 0$, then $\lambda_i(X) \geq \lambda_i(Y)$ for $i = 1, \ldots, \ell$, where $\lambda_i(C)$ is the i-th largest eigenvalue of $C = X, Y$.*

▶ **Proposition 11.** *Let X be the $k \times k$ transition matrix of an ergodic reversible Markov chain with stationary distribution π, and eigenvalues $1 = \lambda_0 > \lambda_1 \geq \cdots \geq \lambda_{k-1}$. Let $X^* = \lim_{t\to\infty} X^t$ be the matrix containing the row vector π on every row. Then the eigenvalues of $\alpha(I - X^*) - \beta(I - X)$ are*

$$\{0\} \cup \{\alpha - \beta(1 - \lambda_i) \mid i = 1, \ldots, k - 1\}.$$

for given constants α and β.

Proof. Since X is the transition matrix of a reversible Markov chain, it holds that the matrix VXV^{-1} is symmetric[9], where $V = \text{diag}(\pi_1^{1/2}, \pi_2^{1/2}, \ldots, \pi_k^{1/2}) = \text{diag}(\sqrt{\pi})$. Using the fact that similar[10] matrices have the same set of eigenvalues we determine the eigenvalues of $\alpha(I - X^*) - \beta(I - X)$ by finding those of

$$V(\alpha(I - X^*) - \beta(I - X))V^{-1} = \alpha(I - \sqrt{\pi}^T\sqrt{\pi}) - \beta(I - VXV^{-1}).$$

Let $\mathbf{1} = (1, 1, 1, \ldots, 1)^T$ denote the all-ones vector. We find

$$VXV^{-1}\sqrt{\pi}^T = VX\mathbf{1} = V\mathbf{1} = \sqrt{\pi}^T,$$

so that $\sqrt{\pi}^T$ is an eigenvector of VXV^{-1} with eigenvalue 1. It then follows that $\sqrt{\pi}^T$ is an eigenvector of $\alpha(I - \sqrt{\pi}^T\sqrt{\pi}) - \beta(I - VXV^{-1})$ with eigenvalue 0. Let $\sqrt{\pi}^T = w_0, w_1, \ldots, w_{k-1}$ be a basis of orthogonal eigenvectors for VXV^{-1} corresponding to eigenvalues $\lambda_1, \ldots, \lambda_{k-1}$ (note that X and VXV^{-1} have the same eigenvalues). It then follows that

$$[\alpha(I - \sqrt{\pi}^T\sqrt{\pi}) - \beta(I - VXV^{-1})]w_i = (\alpha - \beta(1 - \lambda_i))\, w_i$$

because of orthogonality. This completes the proof. ◀

[9] This is the same argument that is used to show that a reversible Markov chain only has real eigenvalues.
[10] Two square matrices A and B are *similar* if there exists an invertible matrix T such that $A = T^{-1}BT$.

Proof of Theorem 2. We first show that all eigenvalues of P are non-negative. Let D be the $|\Omega| \times |\Omega|$ diagonal matrix with $(D)_{xx} = \sqrt{\pi(x)}$. Note that the matrices $D^{-1}P_R D$ are positive semi-definite: they are symmetric because P_R defines a reversible Markov chain on R. The eigenvalues of $D^{-1}P_R D$ are equal to those of P_R, which are all non-negative by assumption. Any non-negative linear combination of positive semi-definite matrices is again positive semi-definite, hence $D^{-1}PD \succeq 0$. Thus P has non-negative eigenvalues. A similar argument holds for P_{heat} and was shown in [12]. In particular, this implies that $\lambda_* = \lambda_1$ and $\lambda_*^{heat} = \lambda_1^{heat}$.

Let

$$Y_R := D^{-1}[\alpha(I_R - \mathbf{1} \cdot \sigma_R) - \beta(I_R - P_R)]D$$

where I_R is defined by $I_R(x,x) = 1$ if $x \in R$ and zero otherwise. The matrix Y_R is symmetric since the matrices $\mathbf{1} \cdot \sigma_R$ and P_R define reversible Markov chains on R. Furthermore its eigenvalues are $\{0\} \cup \{\alpha - \beta(1-\lambda_i) \mid i = 1, \dots, k-1\}$ by Proposition 11 and the fact that similar matrices have the same set of eigenvalues. These eigenvalues are non-negative by assumption, hence Y_R is positive semi-definite. It then follows that the matrix

$$D^{-1}[\alpha(I - P_{heat}) - \beta(I - P)]D = \sum_{a \in \mathcal{L}} \rho(a) \sum_{R \in \mathcal{R}_a} D^{-1}[\alpha(I_R - \mathbf{1} \cdot \sigma_R) - \beta(I_R - P_R)]D$$

is also positive semidefinite. Using Proposition 10 and again the fact that similar matrices have the same set of eigenvalues, it follows that

$$\alpha(1 - \lambda_i^{heat}) \geq \beta(1 - \lambda_i)$$

which finishes the proof. ◀

B Markov chain comparison using Dirichlet forms

In this appendix we include some notes on the comparison framework for Markov chains based on Dirichlet forms and show that, for our setting, it is equivalent to a comparison in terms of positive semidefiniteness. The description is taken from Chapter 13.3 [24].

Let $\mathcal{M}$ be an ergodic, reversible Markov chain on state space Ω with transition matrix P and stationary distribution π. The Dirichlet form for the pair (P, π) is defined by

$$\mathcal{E}(f,h) := \langle (I-P)f, h \rangle_\pi$$

for functions $f, h \in \{g \mid g : \Omega \to \mathbb{R}\}$, where $\langle g_1, g_2 \rangle_\pi = \sum_{x \in \Omega} g_1(x) g_2(x) \pi(x)$. To illustrate the usefulness of Dirichlet forms, consider the following result, which appears, e.g., as Lemma 13.22 in [24].

▶ **Lemma 12.** *Let P and $\tilde{P}$ be reversible transition matrices with stationary distributions π and $\tilde{\pi}$, respectively. If $\tilde{\mathcal{E}}(f,f) \leq \alpha \mathcal{E}(f,f)$ for all $f \in \{g \mid g : \Omega \to \mathbb{R}\}$, then*

$$1 - \tilde{\lambda}_1 \leq \left[\max_{x \in \Omega} \frac{\pi(x)}{\tilde{\pi}(x)}\right] \alpha(1 - \lambda_1),$$

where λ_1 and $\tilde{\lambda}_1$ are resp. the second largest eigenvalue of P and $\tilde{P}$. In particular, if both stationary distributions are the same, we get $1 - \tilde{\lambda}_1 \leq \alpha(1 - \lambda_1)$.

The following proposition relates the Dirichlet form to the use of positive semidefinite matrices, in case both stationary distributions are the uniform distribution over Ω. We can then essentially use the above lemma instead of Proposition 10. We choose to give Proposition 10 as this avoids having to introduce the Dirichlet framework.

▶ **Proposition 13.** *Suppose that π and $\tilde{\pi}$ are both the uniform distribution over Ω. Then $\tilde{\mathcal{E}}(f,f) \leq \alpha\mathcal{E}(f,f)$ is equivalent to*

$$\alpha(I-P) \succeq (I-\tilde{P}).$$

Proof. If both stationary distributions are the uniform distribution over Ω, then the condition

$$\tilde{\mathcal{E}}(f,f) \leq \alpha\mathcal{E}(f,f) \tag{13}$$

is equivalent to

$$f^T(I-\tilde{P})f \leq \alpha f^T(I-P)f$$

where the function f is interpreted as a vector. This in turn is equivalent to stating that $\alpha(I-P) \succeq (I-\tilde{P})$. This follows from the equivalence that $A \succeq 0$ if and only if $x^TAx \geq 0$ for all real-valued vectors x. ◀

Randomness Extraction in AC^0 and with Small Locality

Kuan Cheng[1]
Department of Computer Science, Johns Hopkins University.
kcheng17@jhu.edu.

Xin Li[2]
Department of Computer Science, Johns Hopkins University.
lixints@cs.jhu.edu.

Abstract

Randomness extractors, which extract high quality (almost-uniform) random bits from biased random sources, are important objects both in theory and in practice. While there have been significant progress in obtaining near optimal constructions of randomness extractors in various settings, the computational complexity of randomness extractors is still much less studied. In particular, it is not clear whether randomness extractors with good parameters can be computed in several interesting complexity classes that are much weaker than P.

In this paper we study randomness extractors in the following two models of computation: (1) constant-depth circuits (AC^0), and (2) the local computation model. Previous work in these models, such as [38], [15] and [6], only achieve constructions with weak parameters. In this work we give explicit constructions of randomness extractors with much better parameters. Our results on AC^0 extractors refute a conjecture in [15] and answer several open problems there. We also provide a lower bound on the error of extractors in AC^0, which together with the entropy lower bound in [38, 15] almost completely characterizes extractors in this class. Our results on local extractors also significantly improve the seed length in [6]. As an application, we use our AC^0 extractors to study pseudorandom generators in AC^0, and show that we can construct both cryptographic pseudorandom generators (under reasonable computational assumptions) and unconditional pseudorandom generators for space bounded computation with very good parameters.

Our constructions combine several previous techniques in randomness extractors, as well as introduce new techniques to reduce or preserve the complexity of extractors, which may be of independent interest. These include (1) a general way to reduce the error of strong seeded extractors while preserving the AC^0 property and small locality, and (2) a seeded randomness condenser with small locality.

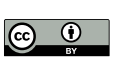
2012 ACM Subject Classification Theory of computation → Expander graphs and randomness extractors, Theory of computation → Pseudorandomness and derandomization

Keywords and phrases Randomness Extraction, AC0, Locality, Pseudorandom Generator

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.37

Related Version A full version of this paper appears in [10], `https://arxiv.org/abs/1602.01530`.

[1] Supported in part by NSF award CCF-1617713.
[2] Supported in part by NSF award CCF-1617713.



1 Introduction

Randomness extractors are functions that transform biased random sources into almost uniform random bits. Throughout this paper, we model biased random sources by the standard model of general weak random sources, which are probability distributions over n-bit strings with a certain amount of min-entropy k.[3] Such sources are referred to as (n, k)-sources. In this case, it is well known that no deterministic extractors can exist for one single weak random source even if $k = n - 1$; therefore seeded randomness extractors were introduced in [32], which allow the extractors to have a short uniform random seed (say length $O(\log n)$). In typical situations, we require the extractor to be *strong* in the sense that the output is close to uniform even given the seed. Formally, we have the following definition.

▶ **Definition 1** ([32]). A function $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a seeded (k, ϵ) extractor if for any (n, k) source X, we have

$$|\mathsf{Ext}(X, U_d) - U_m| \leq \epsilon.$$

Ext is strong if in addition $|(\mathsf{Ext}(X, U_d), U_d) - (U_m, U_d)| \leq \epsilon$, where U_m and U_d are independent uniform strings on m and d bits respectively, and $|\cdot|$ stands for the statistical distance.

Since their introduction, seeded randomness extractors have become fundamental objects in pseudorandomness, and have found numerous applications in derandomization, complexity theory, cryptography and many other areas in theoretical computer science. In addition, through a long line of research, we now have explicit constructions of seeded randomness extractors with almost optimal parameters (e.g., [17]). However, the complexity of randomness extractors is still much less studied and understood. For example, while in general explicit constructions of randomness extractors can be computed in polynomial time of the input size, some of the known constructions are actually more explicit than that. These include for example extractors based on universal hashing [8], and Trevisan's extractor [36], which can be computed by highly uniform constant-depth circuits of polynomial size with parity gates. Thus a main question one can ask is: can we do better and construct good randomness extractors with very low complexity?

This question is interesting not just by its own right, but also because such extractors, as building blocks, can be used to potentially reduce the complexity of other important objects. In this paper we study this question and consider the parallel and local complexity of randomness extractors.

The parallel-AC^0 model. The hierarchy of NC and AC circuits are standard models for parallel computation. It is easy to see that the class of NC^0 or even ℓ-local functions for small ℓ , which correspond to functions where each output bit depends on at most ℓ input bits (including both the weak source and the seed), cannot compute strong extractors (since one can just fix ℓ bits of the source). Thus, a natural relaxation is to consider the class AC^0, which refers to the family of polynomial-size and constant-depth circuits with unbounded fan-in gates. Note that although we have strong lower bounds here for explicit functions, it is still not clear whether some important objects, such as randomness extractors and pseudorandom generators, can be computed in AC^0 with good parameters. Thus the study of this question also helps us better understand the power of this class.

[3] A probability distribution is said to have min-entropy k if the probability of getting any element in the support is at most 2^{-k}.

Viola [38] was the first to consider this question, and his result was generalized by Goldreich et al. [15] to show that for strong seeded extractors, even extracting a single bit is impossible if $k < n/\mathsf{poly}(\log n)$. When $k \geq n/\mathsf{poly}(\log n)$, Goldreich et al. showed how to extract $\Omega(\log n)$ bits using $O(\log n)$ bits of seed, or more generally how to extract $m < k/2$ bits using $O(m)$ bits of seed. Note that the seed length is longer than the output length.[4] When the extractor does not need to be strong, they showed that extracting $r + \Omega(r)$ bits using r bits of seed is impossible if $k < n/\mathsf{poly}(\log n)$; while if $k \geq n/\mathsf{poly}(\log n)$ one can extract $(1 + c)r$ bits for some constant $c > 0$, using r bits of seed. All the positive results here have error $1/\mathsf{poly}(n)$.

Therefore, a natural and main open problem left in [15] is whether one can construct randomness extractors in AC^0 with shorter seed and longer output. Specifically, [15] asks if one can extract more than $\mathsf{poly}(\log n)r$ bits in AC^0 using a seed length $r = \Omega(\log n)$, when $k \geq n/\mathsf{poly}(\log n)$. In [15] the authors conjectured that the answer is negative. Another open question is to see if one can achieve better error, e.g., negligible error instead of $1/\mathsf{poly}(n)$.

Goldreich et al. [15] also studied deterministic extractors for bit-fixing sources, and most of their effort went into extractors for oblivious bit-fixing sources (although they also briefly studied non-oblivious bit-fixing sources). An (n, k)-oblivious bit-fixing source is a string of n bits such that some unknown k bits are uniform, while the other $n - k$ bits are fixed. Extractors for such sources are closely related to exposure-resilient cryptography [7, 24]. In this case, a standard application of Håstad's switching lemma [18] implies that it is impossible to construct extractors in AC^0 for bit-fixing sources with min-entropy $k < n/\mathsf{poly}(\log n)$. The main result in [15] is a theorem which shows the *existence* of deterministic extractors in AC^0 for min-entropy $k \geq n/\mathsf{poly}(\log n)$ that output $k/\mathsf{poly}(\log n)$ bits with error $2^{-\mathsf{poly}(\log n)}$. We emphasize that this is an existential result, and [15] did not give any explicit constructions of such extractors.

The local model. Another relaxation, introduced by Bogdanov and Guo [6], is the notion of sparse extractor families. These are families of functions for which each function in the family has a small number of overall input-output dependencies (referred to as the sparsity, meaning that the input-output dependency graph is sparse), while taking a random function from the family serves as a randomness extractor. Such extractors can be used generally in situations where hashing is used and preserving small input-output dependencies is needed. As an example, the authors in [6] used such extractors to obtain a transformation of non-uniform one-way functions into non-uniform pseudorandom generators that preserves output locality.

In this paper, we consider the condition of the family being ℓ-local, which is a worst case notion rather than the average case notion of sparsity. Furthermore, we will focus on the case of strong extractor families. Note that a strong extractor family is equivalent to a strong seeded extractor, since the randomness used to choose a function from the family can be included in the seed. Thus, we study strong seeded extractors with small locality, i.e., for any fixing of the seed, each output bit depends on at most ℓ input bits.

Note that an extractor family with m output bits and locality ℓ is automatically an ℓm-sparse extractor family. Conversely, if an extractor is s-sparse, then half of its output bits depend on at most $2s/m$ input bits, so by removing half of the output bits one could obtain locality $2s/m$; a technical point here is that one may need to drop different output bits depending on the seed, but this does not affect the error of the extractor.

[4] They also showed how to extract $\mathsf{poly}(\log n)$ bits using an $O(\log n)$ bit seed, but the error of the extractor becomes $1/\mathsf{poly}(\log n)$.

The authors of [6] gave a construction of a strong extractor family for all entropy k with output length $m \leq k$, error ϵ, and sparsity $O(n \log(m/\epsilon) \log(n/m))$, which corresponds to locality $O(\frac{n}{m} \log(\frac{m}{\epsilon}) \log(\frac{n}{m})) = \Omega(n/k \log(n/\epsilon))$ whenever $k \leq n/2$. They also showed that such sparsity is necessary whenever $n^{0.99} \leq m \leq n/6$ and ϵ is a constant. However, the main drawback of the construction in [6] is that the family size is quite large. Indeed the family size is 2^{nm}, which corresponds to a seed length of at least nm.[5] Therefore, a main open problem in [6] is to reduce the size of the family (or, equivalently, the seed length).

De and Trevisan [11] obtained a strong extractor for (n, k) sources such that for any fixing of the seed, each bit of the extractor's output only depends on $\mathsf{poly}(\log n)$ bits of the source. However, their construction only works for $k = \delta n$ where δ is any constant. Their extractor has seed length $d = O(\log n)$ and outputs $k^{\Omega(1)}$ bits, but the error is only $n^{-\alpha}$ for a small constant $0 < \alpha < 1$.

It is also worthwhile to compare our definition of a strong extractor family with small locality to the definition of t-local extractors given by Vadhan [37]. For a t-local extractor, one requires that for any fixing of the seed r, the output of the function $\mathsf{Ext}(x, r)$ *as a whole* depends on only t bits of x. In contrast, our definition requires that *each output bit* of the function $\mathsf{Ext}(x, r)$ depends on at most ℓ bits of x. It can be seen that a strong extractor with sparsity t is automatically a t-local extractor, but the converse may not be true: a t-local extractor may have locality t and sparsity up to mt. By a lower bound in [37], the parameter t in local-extractors is at least $\Omega(nm/k)$, which is larger than m and matches the sparsity in [6] and our results up to polylogarithmic factors. In this sense, our definition of extractor with small locality is stronger than t-local extractors. Furthermore, the construction of t-local extractors in [37], which uses the sample-then-extract approach, only works for large min-entropy (at least $k > \sqrt{n}$); while our goal here is to construct strong extractor families even for very small min-entropy, with locality $\ell \ll m$.

Our results

As in [38, 15], in this paper we obtain both negative results and positive results about randomness extraction in AC^0. While the negative results in [38, 15] provide lower bounds on the entropy required for AC^0 extractors, our negative results provide lower bounds on the error such extractors can achieve. We show that such extractors (both seeded extractors and deterministic extractors for bit-fixing sources) cannot achieve error better than $2^{-\mathsf{poly}(\log n)}$, even if the entropy of the sources is quite large. Specifically, we have

▶ **Theorem 2.** *(General weak source) If* $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *is a strong* $(k = n - 1, \epsilon)$*-extractor that can be computed by* AC^0 *circuits of depth* $\mathtt{dth}$ *and size* s*, then* $\epsilon = 2^{-(O(\log s))^{\mathtt{dth}-1} \log(n+d)}$.

(Bit-fixing source) There is a constant $c > 1$ *such that if* $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$ *is a* (k, ϵ)*-extractor for oblivious bit-fixing sources with* $k = n - (c \log s)^{\mathtt{dth}-1}$*, that can be computed by* AC^0 *circuits of depth* $\mathtt{dth}$ *and size* s*, then* $\epsilon = 2^{-(O(\log s))^{\mathtt{dth}-1} \log n}$. [6]

Thus, our results combined with the lower bounds on the entropy requirement in [38, 15] almost completely characterize the power of randomness extractors in AC^0.

We now turn to our positive results. As our first contribution, we show that the authors' conjecture about seeded AC^0 extractors in [15] is false. We give explicit constructions of *strong* seeded extractors in AC^0 with much better parameters. This in particular answers open problems 8.1 and 8.2 in [15]. To start with, we have the following theorem.

[5] In fact, the seed length is even larger since the seed is used to sample from a non-uniform distribution.
[6] This holds even if we allow Ext to have a uniform random seed, see in the full version [10].

▶ **Theorem 3.** *For any constant $c \in \mathbb{N}$, any $k = \Omega(n/\log^c n)$ and any $\epsilon = 1/\mathsf{poly}(n)$, there exists an explicit construction of a strong (k,ϵ)-extractor $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ that can be computed by an AC^0 circuit of depth $c+10$, where $d = O(\log n)$, $m = k^{\Omega(1)}$ and the extractor family has locality $O(\log^{c+5} n)$.*

Note that the depth of the circuit is almost optimal, within an additive $O(1)$ factor of the lower bound given in [15]. In addition, our construction is also a family with locality only $\mathsf{poly}(\log n)$. Note that the seed length $d = O(\log n)$ is (asymptotically) optimal, while the locality beats the one obtained in [6] (which is $O(n/m \log(m/\epsilon)\log(n/m)) = n^{\Omega(1)}$) and is within a $\log^4 n$ factor to $O(n/k \log(n/\epsilon))$.

Our result also improves that of De and Trevisan [11], even in the high min-entropy case, as our error can be any $1/\mathsf{poly}(n)$ instead of just $n^{-\alpha}$ for some constant $0 < \alpha < 1$. Moreover, our seed length remains $O(\log n)$ even for $k = n/\mathsf{poly}(\log n)$, while in this case the extractor in [11] has seed length $\mathsf{poly}(\log n)$.

Next, we can boost our construction to reduce the error and extract almost all the entropy. We have

▶ **Theorem 4.** *For any constant $\gamma \in (0,1)$, $a, c \in \mathbb{N}$, any $k = \delta n = \Omega(n/\log^c n)$, $\epsilon = 1/2^{O(\log^a n)}$, there exists an explicit strong (k,ϵ)-extractor $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ in AC^0 with depth $O(a+c+1)$ where $d = O((\log n + \frac{\log(n/\epsilon)\log(1/\epsilon)}{\log n})/\delta)$, $m = (1-\gamma)k$.*

As our second contribution, we give *explicit* deterministic extractors in AC^0 for oblivious bit-fixing sources with entropy $k \geq n/\mathsf{poly}(\log n)$, which output $(1-\gamma)k$ bits with error $2^{-\mathsf{poly}(\log n)}$. This is in contrast to the non-explicit existential result in [15]. Further, the output length and error of our extractor are almost optimal, while the output length in [15] is only $k/\mathsf{poly}(\log n)$. Specifically, we have

▶ **Theorem 5.** *For any constant $a, c \in \mathbb{N}$ and any constant $\gamma \in (0,1]$, there exists an explicit deterministic $(k = \Omega(n/\log^a n), \epsilon = 2^{-\log^c n})$-extractor $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^{(1-\gamma)k}$ that can be computed by AC^0 circuits of depth $O(a+c+1)$, for any (n,k)-bit-fixing source.*

For sparse extractor families, we can reduce the error of Theorem 3 while keeping the locality small.

▶ **Theorem 6.** *There exists a constant $\alpha \in (0,1)$ such that for any $k \geq \frac{n}{\mathsf{poly}(\log n)}$ and $\epsilon \geq 2^{-k^\alpha}$, there exists an explicit construction of a strong (k,ϵ)-extractor $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$, with $d = O(\log n + \frac{\log(n/\epsilon)\log(1/\epsilon)}{\log n})$, $m = k^{\Omega(1)}$ and locality $\log^2(1/\epsilon)\mathsf{poly}(\log n)$.*

We also give strong extractor families with small locality for min-entropy k as small as $\log^2 n$. Our approach is to first condense it into another weak source with constant entropy rate. For this purpose we introduce the following definition of a (strong) randomness condenser with small locality.

▶ **Definition 7.** A function $\mathsf{Cond} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{n_1}$ is a strong (n,k,n_1,k_1,ϵ)-condenser if for every (n,k)-source X and independent uniform seed $R \in \{0,1\}^d$, $R \circ \mathsf{Cond}(X,R)$ is ϵ-close to $R \circ D$, where D is a distribution on $\{0,1\}^{n_1}$ such that for any $r \in \{0,1\}^d$, we have that $D|_{R=r}$ is an (n_1,k_1)-source. We say the condenser family has locality ℓ if for every fixing of $R = r$, the function $\mathsf{Cond}(.,r)$ can be computed by an ℓ-local function.

We now have the following theorem.

▶ **Theorem 8.** *For any $k \geq \log^2 n$, there exists a strong $(n, k, t = 10k, 0.08k, \epsilon)$-condenser $\mathsf{Cond} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^t$ with $d = O(k)$, $\epsilon = 2^{-\Omega(k)}$ and locality $O(\frac{n}{k} \log n)$.*

Combining the condenser with our previous extractors, we get strong extractor families with small locality for any min-entropy $k \geq \log^2 n$. Specifically, we have

▶ **Theorem 9.** *There exits a constant $\alpha \in (0,1)$ such that for any $k \geq \log^2 n$, any constant $\gamma \in (0,1)$ and any $\epsilon \geq 2^{-k^\alpha}$, there exists a strong (k, ϵ)-extractor $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$, where $d = O(k), m = (1-\gamma)k$ and the extractor family has locality $\frac{n}{k} \log^2(1/\epsilon)(\log n)\mathsf{poly}(\log k)$.*

In the above two extractors, our seed length is still much better than that of [6]. However, our locality becomes slightly worse.

Related work, independent work and further results

The work of Dziembowski and Maurer [12] also gave extractors with small locality. However, the model of the weak source studied there is uniform random bits subject to a bounded leakage, which is more restrictive than the model of general weak random sources we consider here. In particular, the analysis of [12], which uses a guessing game, may not work for general weak random sources. A recent independent work by Papakonstantinou et. al [34] used similar techniques as [12] and gave constructions of seeded extractors in the multi-stream model [16]. Their main motivation and result is an extractor that can extract $\Omega(k)$ bits from any (n, k) source with $k = \Omega(n)$, using $O(\log n \log(n/\epsilon))$ bits of seed together with two streams, $O(\log\log(n/\epsilon))$ passes and $O(\log(n/\epsilon))$ space. However, it turns out that their construction can also be realized in AC^0 and also has the property of small locality. Specifically, for an (n, k) source with $k = \delta n = n/\mathsf{poly}\log(n)$ and error $\epsilon = 2^{-\mathsf{poly}\log(n)}$, their construction gives an AC^0 extractor with seed length $O(\frac{1}{\delta^{O(1)}} \log n \log(n/\epsilon))$ and locality $O(\frac{1}{\delta^{O(1)}} \log(n/\epsilon))$. Their construction, which is based on randomly re-bucketing the bits of the source into blocks and arguing this results in a block source, can be viewed as orthogonal to our construction, which is based on hardness amplification. Compared to our results (Theorem 4 and Theorem 6), they have a better dependence on ϵ but we have a better dependence on δ and n. For very small entropy (e.g., $k = \log^2 n$), we can first use our condenser and then apply their construction, which will give an extractor with seed length $O(k)$, output length $\Omega(k)$ and locality $O(\frac{n}{k} \log n \log(k/\epsilon))$.

Applications to pseudorandom generators in AC^0

Like extractors, pseudorandom generators are also fundamental objects in the study of pseudorandomness, and constructing "more explicit" pseudorandom generators is another interesting question that has gained a lot of attention. A pseudorandom generator (or PRG for short) is an efficient deterministic function that maps a short random seed into a long output that looks uniform to a certain class of distinguishers.

▶ **Definition 10.** A function $G : \{0,1\}^n \to \{0,1\}^m$ is a pseudorandom generator for a class $\mathcal{C}$ of Boolean functions with error ϵ, if for every function $A \in \mathcal{C}$, we have that

$$|\Pr[A(U_m) = 1] - \Pr[A(G(U_n)) = 1]| \leq \epsilon.$$

Here we mainly consider two kinds of pseudorandom generators, namely cryptographic PRGs, which are necessarily based on computational assumptions; and unconditional PRGs, most notably PRGs for space bounded computation.

Standard cryptographic PRGs (i.e., PRGs that fool polynomial time computation or polynomial size circuits with negligible error) are usually based on one-way functions (e.g., [19]), and can be computed in polynomial time. However, more explicit PRGs have also been considered in the literature, for the purpose of constructing more efficient cryptographic protocols. Impagliazzo and Naor [23] showed how to construct such a PRG in AC^0, which stretches n bits to $n + \log n$ bits. Their construction is based on the assumed intractability of the subset sum problem. On the other hand, Viola [39] showed that there is no black-box PRG construction with linear stretch in AC^0 from one-way functions. Thus, to get such stretch one must use non black-box constructions.

In [3, 4], Applebaum et al. showed that the existence of cryptographic PRGs in NC^0 with sub-linear stretch follows from a variety of standard assumptions, and they constructed a cryptographic PRG in NC^0 with linear stretch based on a specific intractability assumption related to the hardness of decoding sparsely generated linear codes. In [2], Applebaum further constructed PRG *collections* (i.e., a family of PRG functions) with linear stretch and polynomial stretch based on the assumption of one-wayness of a variant of the random local functions proposed by Goldreich [14].

In the case of unconditional PRGs, for $d \geq 5$ Mossel et al. [29] constructed d-local PRGs with output length $n^{\Omega(d/2)}$ that fool all linear tests with error $2^{-n^{\frac{1}{2\sqrt{d}}}}$, which were used by Applebaum et al. [3] to give a 3-local PRG with linear stretch that fools all linear tests. In the same paper, Applebaum et al. also gave a 3-local PRG with sub linear stretch that fools sublinear-space computation. Thus, it remains to see if we can construct better PRGs (cryptographic or unconditional) in NC^0 or AC^0 with better parameters.

Our PRGs. We show that under reasonable computational assumptions, we can construct very good cryptographic PRGs in AC^0 (e.g. with polynomial stretch and negligible error). In addition, we show that we can construct very good unconditional PRGs for space bounded computation in AC^0 (e.g., with polynomial stretch).

We first give explicit cryptographic PRGs in AC^0 based on the one-wayness of random local functions, the same assumption as used in [2]. To state the assumption we first need the following definitions.

▶ **Definition 11** (Hypergraphs [2])**.** An (n, m, d) hypergraph is a graph over n vertices and m hyperedges each of cardinality d. For each hyperedge $S = (i_0, i_1, \ldots, i_{d-1})$, the indices $i_0, i_1, \ldots, i_{d-1}$ are ordered. The hyperedges of G are also ordered. Let G be denoted as $([n], S_0, S_1, \ldots, S_{m-1})$ where for $i = 0, 1, \ldots, m-1$, S_i is a hyperedge.

▶ **Definition 12** (Goldreich's Random Local Function [14])**.** Given a predicate $Q : \{0,1\}^d \to \{0,1\}$ and an (n, m, d) hypergraph $G = ([n], S_0, \ldots, S_{m-1})$, the function $f_{G,Q} : \{0,1\}^n \to \{0,1\}^m$ is defined as follows: for input x, the ith output bit of $f_{G,Q}(x)$ is $f_{G,Q}(x)_i = Q(x_{S_i})$.

For $m = m(n)$, the function collection $F_{Q,n,m} : \{0,1\}^s \times \{0,1\}^n \to \{0,1\}^m$ is defined via the mapping $(G, x) \to f_{G,Q}(x)$, where G is sampled randomly by the s bits and x is sampled randomly by the n bits.

For every $k \in \{0,1\}^s$, we also denote $F(k, \cdot)$ as $F_k(\cdot)$.

▶ **Definition 13** (One-wayness of a Collection of Functions)**.** For $\epsilon = \epsilon(n) \in (0, 1)$, a collection of functions $F : \{0,1\}^s \times \{0,1\}^n \to \{0,1\}^m$ is an ϵ-one-way function if for every efficient adversary A which outputs a list of $\mathsf{poly}(n)$ candidates and for sufficiently large n's, we have that

$$\Pr_{k,x,y=F_k(x)}[\exists z \in A(k, y), z' \in F_k^{-1}(y), z = z'] < \epsilon,$$

where k and x are independent and uniform.

We now have the following theorem.

▶ **Theorem 14.** *For any d-ary predicate Q, if the random local function $F_{Q,n,m}$ is δ-one-way for some constant $\delta \in (0,1)$, then we have the following results.*

1. *If there exists a constant $\alpha > 0$ such that $m \geq (1+\alpha)n$, then for any constant $c > 1$, there exists an explicit cryptographic PRG $G : \{0,1\}^r \to \{0,1\}^t$ in AC^0, where $t \geq cr$ and the error is negligible*[7]*.*
2. *If there exists a constant $\alpha > 0$ such that $m \geq n^{1+\alpha}$, then for any constant $c > 1$ there exists an explicit cryptographic PRG $G : \{0,1\}^r \to \{0,1\}^t$ in AC^0, where $t \geq r^c$ and the error is negligible.*

As noted in [2], there are several evidence supporting this assumption. In particular, current evidence is consistent with the existence of a δ-one-way random local function $F_{Q,n,m}$ with $m \geq n^{1+\alpha}$ for some constant $\alpha > 0$.

Compared to the constructions in [2], our construction is in AC^0 instead of NC^0. However, our construction has the following advantages.

- We construct a standard PRG instead of a PRG collection, where the PRG collection is a family of functions and one needs to randomly choose one function before any application.
- The construction of a PRG with polynomial stretch in [2] can only achieve polynomially small error, and for negligible error one needs to assume that the random local function cannot be inverted by any adversary with slightly super polynomial running time. Our construction, on the other hand, achieves negligible error while only assuming that the random local function cannot be inverted by any adversary that runs in polynomial time.

Next we give an explicit PRG in AC^0 with polynomial stretch, that fools space bounded computation. It is a straight forward application of our AC^0-extractor to the Nisan-Zuckerman PRG [32].

▶ **Theorem 15.** *For every constant $c \in \mathbb{N}$ and every $m = m(s) = \mathsf{poly}(s)$, there is an explicit PRG $g : \{0,1\}^{r=O(s)} \to \{0,1\}^m$ in AC^0, such that for any randomized algorithm A using space s,*

$$|\Pr[A(g(U_r)) = 1] - \Pr[A(U_m) = 1]| = \epsilon \leq 2^{-\Theta(\log^c s)},$$

where U_r is the uniform distribution of length r, U_m is the uniform distribution of length m.

Compared to the Nisan-Zuckerman PRG [32], our PRG is in $\mathsf{AC0}$, which is more explicit. On the other hand, our error is $2^{-\Theta(\log^c s)}$ for any constant $c > 0$ instead of being exponentially small as in [32]. It is a natural open problem to see if we can reduce the error to exponentially small. We note that this cannot be achieved by simply hoping to improve the extractor, since our negative result shows that seeded extractors in AC^0 cannot achieve error better than $2^{-\mathsf{poly}(\log n)}$.

2 Lower bounds for error parameters of AC^0 extractors

Our negative results about the error of AC^0 extractors follow by a simple application of Fourier analysis and the well known spectrum concentration theorem of AC^0 functions [27]. We present it in Appendix A.

[7] The error $\epsilon : \mathbb{N} \to [0,1]$ is negligible if $\epsilon(n) = n^{-\omega(1)}$.

3 Constructions of AC^0 extractors for general weak sources

We now briefly describe our positive results. For details of constructions and proofs, please refer to the full version [10]. We will extensively use the following two facts: the parity and inner product over $\mathsf{poly}(\log n)$ bits can be computed by AC^0 circuits of size $\mathsf{poly}(n)$; in addition, any Boolean function on $O(\log n)$ bits can be computed by a depth-2 AC^0 circuit of size $\mathsf{poly}(n)$.

3.1 The basic construction

All our constructions are based on a basic construction of a strong extractor in AC^0 for any $k \geq \frac{n}{\mathsf{poly}(\log n)}$ with seed length $d = O(\log n)$ and error $\epsilon = n^{-\Omega(1)}$. This construction is a a modification of the Impagliazzo-Wigderson pseudorandom generator [21], interpreted as a randomness extractor in the general framework found by Trevisan [36]. The IW-generator first takes a Boolean function on $\log n$ bits, applies a series of hardness amplifications to get another Boolean function on $O(\log n)$ bits, and then uses the Nisan-Wigderson generator [31] together with the new Boolean function. The hardness amplification consists of three steps: the first step, developed by Babai et al. [5], is to obtain a mild average-case hard function from a worst-case hard function; the second step involves a constant number of substeps, with each substep amplifying the hardness by using Impagliazzo's hard core set theorem [22]; the third step, developed by Impagliazzo and Wigderson [21], uses a derandomized direct-product generator to obtain a function that can only be predicted with exponentially small advantage.

Trevisan [36] showed that given an (n, k)-source X, if one regards the n bits of X as the truth table of the initial Boolean function on $\log n$ bits and applies the IW-generator, then by setting parameters appropriately one obtains an extractor. The reason is that for any $x \in \mathsf{supp}(X)$ that makes the output of the extractor fail a certain statistical test T, one can "reconstruct" x by showing that it can computed by a small size circuit, when viewing x as the truth table of the function with T gates. Thus the number of such bad elements $x \in \mathsf{supp}(X)$ is upper bounded by the total number of such circuits. This extractor works for any min-entropy $k \geq n^{\alpha}$.

However, this extractor itself is not in AC^0 (which is not surprising since it can handle min-entropy $k \geq n^{\alpha}$). Thus, at least one of the steps in the construction of the IW-generator/extractor is not in AC^0. By carefully examining each step one can see that the only step not in AC^0 is actually the first step of hardness amplification (This was also pointed out by [38]). Indeed, all the other steps of hardness amplification are essentially doing the same thing: obtaining a function f' on $O(\log n)$ bits from another function f on $O(\log n)$ bits, where the output of f' is obtained by taking the inner product over two $O(\log n)$ bit strings s and r. In addition, s is obtained directly from part of the input of f', while r is obtained by using the other part of the input of f' to generate $O(\log n)$ inputs to f and concatenating the outputs. All of these steps can be done in AC^0, assuming f is in AC^0 (note that f here depends on X).

We therefore modify the IW-generator by removing the first step of hardness amplification, and start with the second step of hardness amplification with the source X as the truth table of the initial Boolean function. Thus the initial function f can be computed by using the $\log n$ input bits to select a bit from X, which can be done in AC^0. Therefore the final Boolean function f' can be computed in AC^0. The last step of the construction, which applies the NW-generator, is just computing f' on several blocks of size $O(\log n)$, which certainly is in AC^0. This gives our basic extractor in AC^0.

The analysis is again similar to Trevisan's argument [36]. However, since we have removed the first step of hardness amplification, now for any $x \in \mathsf{supp}(X)$ that makes the output of the extractor to fail a certain statistical test T, we cannot obtain a small circuit that *exactly computes* x. On the other hand, we can obtain a small circuit that can *approximate* x well, i.e., can compute x correctly on $1-\gamma$ fraction of inputs for some $\gamma = 1/\mathsf{poly}(\log n)$. We then argue that the total number of strings within relative distance γ to the outputs of the circuit is bounded, and therefore combining the total number of possible circuits we can again get a bound on the number of such bad elements in $\mathsf{supp}(X)$. A careful analysis shows that our extractor works for any min-entropy $k \geq n/\mathsf{poly}(\log n)$. However, to keep the circuit size small we have to set the output length to be small enough, i.e., n^{α} and set the error to be large enough, i.e., $n^{-\beta}$.

In Appendix B, we describe more details about the basic construction.

3.1.1 Error reduction

We now describe how we reduce the error of the extractor. We will borrow some techniques from the work of Raz et al. [35], where the authors showed a general way to reduce the error of strong seeded extractors. However, the techniques in Raz et al. [35] do not preserve the AC^0 property, thus our techniques are significantly different from theirs. Nevertheless, our starting point is a lemma from [35], which roughly says the following: given any strong seeded (k,ϵ)-extractor Ext with seed length d and output length m, then for any $x \in \{0,1\}^n$ there exists a set $G_x \subset \{0,1\}^d$ of density $1-O(\epsilon)$, such that if X is a source with entropy slightly larger than k, then the distribution $\mathsf{Ext}(X, G_X)$ is very close to having min-entropy $m - O(1)$. Here $\mathsf{Ext}(X, G_X)$ is the distribution obtained by first sampling x according to X, then sampling r uniformly in G_x and outputting $\mathsf{Ext}(x, r)$.

Giving this lemma, we can apply our basic AC^0 extractor with error $\epsilon = n^{-\beta}$ for some t times, each time with fresh random seed, and then concatenate the outputs. By the above lemma, the concatenation is roughly $(O(\epsilon))^t$-close to a source such that one of the output has min-entropy $m - O(1)$ (i.e., a somewhere high min-entropy source). By choosing t to be a large enough constant the $(O(\epsilon))^t$ can be smaller than any $1/\mathsf{poly}(n)$. We now describe how to extract from the somewhere high min-entropy source with error smaller than any $1/\mathsf{poly}(n)$, in AC^0. This is where our construction differs significantly from [35], as there one can simply apply a good extractor for constant entropy rate.

Assume that we have an AC^0 extractor Ext' that can extract from $(m, m-\sqrt{m})$-sources with error any $\epsilon' = 1/\mathsf{poly}(n)$ and output length $m^{1/3}$. Then we can extract from the somewhere high min-entropy source as follows. We use Ext' to extract from each row of the source with fresh random seed, and then compute the XOR of the outputs. We claim the output is $(2^{-m^{\Omega(1)}} + \epsilon')$-close to uniform. To see this, assume without loss of generality that the i'th row has min-entropy $m - O(1)$. We can now fix the outputs of all the other rows, which has a total size of $tm^{1/3} \ll \sqrt{m}$ as long as t is small. Thus, even after the fixing, with probability $1 - 2^{-m^{\Omega(1)}}$, we have that the i'th row has min-entropy at least $m - \sqrt{m}$. By applying Ext' we know that the XOR of the outputs is close to uniform.

What remains is the extractor Ext'. To construct it we divide the source with length m sequentially into $m^{1/3}$ blocks of length $m^{2/3}$. Since the source has min-entropy $m - \sqrt{m}$, this forms a block source such that each block roughly has min-entropy at least $m^{2/3} - \sqrt{m}$ conditioned on the fixing of all previous ones. We can now take a strong extractor Ext'' in AC^0 with seed length $O(\log n)$ and use the same seed to extract from all the blocks, and concatenate the outputs. It suffices to have this extractor output one bit for each block. Such AC^0 extractors are easy to construct since each block has high min-entropy rate (i.e., $1 - o(1)$). For example, we can use the extractors given by Goldreich et al. [15].

It is straightforward to check that our construction is in AC^0, as long as the final step of computing the XOR of t outputs can be done in AC^0. For error $1/\mathsf{poly}(n)$, it suffices to take t to be a constant and the whole construction is in AC^0, with seed length $O(\log n)$. We can even take t to be $\mathsf{poly}(\log n)$, which will give us error $2^{-\mathsf{poly}(\log n)}$.

3.1.2 Increasing output length

The error reduction step reduces the output length from m to $m^{1/3}$, which is still $n^{\Omega(1)}$. We can increase the output length by using a standard boosting technique as that developed by Nisan and Zuckerman [32, 40]. Specifically, we first use random bits to sample several blocks from the source, using a sampler in AC^0. We then apply our AC^0 extractor on the blocks backwards, and use the output of one block as the seed to extract from the previous block. When doing this we divide the seed into blocks each with the same length as the seed of the AC^0 extractor, apply the AC^0 extractor using each block as the seed, and then concatenate the outputs. This way each time the output will increase by a factor of $n^{\Omega(1)}$. Thus after a constant number of times it will become say $\Omega(k)$. Since each step is computable in AC^0, the whole construction is still in AC^0.

4 Explicit AC^0 extractors for bit-fixing sources

Our explicit AC^0 extractors for (oblivious) bit-fixing sources follow the high-level idea in [15]. Specifically, we first reduce the oblivious bit-fixing source to a non-oblivious bit-fixing source, and then apply an extractor for non-oblivious bit-fixing sources. This approach is natural in the sense that the best known extractors for oblivious bit-fixing sources (e.g., parity or [24]) can both work for small entropy and achieve very small error. Thus by the negative results in [15] and our paper, none of these can be in AC^0. However, extractors for non-oblivious bit-fixing sources are equivalent to resilient functions, and there are well known resilient functions in AC^0 such as the Ajtai-Linial function [1].

The construction in [15] is not explicit, but only existential for two reasons. First, at that time the Ajtai-Linial function is a random function, and there was no explicit construction matching it. Second, the conversion from oblivious-bit fixing source to non-oblivious bit-fixing source in [15] is to multiply the source by a random matrix, for which the authors of [15] showed its existence but were not able to give an explicit construction. Now, the first obstacle is solved by recent explicit constructions of resilient functions in AC^0 that essentially match the Ajtai-Linial function ([9, 28, 26]). Here we use the extractor in [26] that can output many bits. For the second obstacle, we notice that the extractors for non-oblivious bit-fixing sources in [9, 26] do not need the uniform bits to be independent, but rather only require $\mathsf{poly}(\log N)$-wise independence if N is the length of the source.

By exploiting this property, we can give an explicit construction of the matrix used to transfrom the original oblivious bit-fixing source. Our construction is natural and simpler than that in [15], in the sense that it is a matrix over F_2 while the matrix in [15] uses fields of larger size. Specifically, we will take a seeded extractor and view it as a bipartite graph with $N = n^{O(1)}$ vertices on the left, n vertices on the right and left degree $d = \mathsf{poly}(\log N) = \mathsf{poly}(\log n)$. We identify the right vertices with the n bits of the bit-fixing source, and for each left vertex we obtain a bit which is the parity of its neighbors. The new non-oblivious bit-fixing source is the N bit source obtained by concatenating the left bits.

Now suppose the original source has entropy $k = \delta n$ for some $\delta \geq 1/\mathsf{poly}(\log n)$, and let T denote the unfixed bits. A standard property of the seeded extractor implies that most of the left vertices have a good fraction of neighbors in T (i.e., an extractor is a good

sampler), so that each left bit obtained from these vertices is uniform. Next we would like to argue that they are $\mathsf{poly}(\log N)$-wise independent. For this we require the seeded extractor to have a stronger property: that it is a *design extractor* as defined by Li [25]. Besides being an extractor itself, a design extractor requires that any pair of left vertices have a small intersection of neighbors. Assuming this property, it is easy to show that if we take any small subset S of the "good" left vertices, then there is a bit in T that is only connected to a single vertex in S (i.e., a unique neighbor). Thus the XOR of any small enough subset of the "good" left bits is uniform, which indicates that they are some t-wise independent. Several explicit constructions of design extractors were given in [25], and for our applications it suffices to use a simple greedy construction. By adjusting the parameters, we can ensure that $t = \mathsf{poly}(\log N)$ which is enough for applying the extractor in [26]. In addition, the degree $d = \mathsf{poly}(\log N)$ so the parity of d bits can be computed in AC^0.

Once we have the basic extractor, we can use the same techniques as in [15] to reduce the error, and use the techniques by Gabizon et al.[13] to increase the output length (this is also done in [15]). Note that the techniques in [13] require a seeded extractor. In order for the whole construction to be in AC^0, we use our previously constructed seeded extractor in AC^0 which can output $(1-\gamma)k$ bits. Thus we obtain almost optimal explicit AC^0 extractors for oblivious bit-fixing sources. In contrast, the seeded extractor used in [15] only outputs $k/\mathsf{poly}(\log n)$ bits, and thus their (non-explicit) AC^0 extractor for oblivious bit-fixing sources also only outputs $k/\mathsf{poly}(\log n)$ bits.

5 Extractors with small locality for low entropy

Our basic extractor (Theorem 3) also enjoys the property of small locality, but it only works for large entropy. To get constructions for small min-entropy, we adapt the techniques in [6]. There the authors constructed a strong extractor family with small sparsity by randomly sampling an $m \times n$ matrix M and outputting MX, where X is the (n,k)-source. Each entry in M is independently sampled according to a Bernoulli distribution, and thus the family size is 2^{nm}. We derandomize this construction by sampling the second row to the last row using a random walk on an expander graph, starting from the first row. For the first row, we observe that the process of generating the entries and doing inner product with X can be realized by read-once small space computation, thus we can sample the first row using the output of a pseudorandom generator for space bounded computation (e.g., Nisan's generator [30]). We show that this gives us a very good condenser with small locality, i.e., Theorem 8. Combining the condenser with our previous extractors we then obtain strong extractor families with small locality.

6 Applications to pseudorandom generators

For cryptographic pseudorandom generators, we mainly adapt the approach of Applebaum [2], to the AC^0 setting. The construction of cryptographic pseudorandom generator *families* in [2] is based on random local functions. Specifically, given a random bipartite graph with n left vertices, m right vertices and right degree d (think of d as a constant), and a suitable predicate P on d bits, Applebaum showed that based on a conjecture on random local one-way functions, the m output bits obtained by applying P to the m subsets of input bits corresponding to the hyper edges give a distribution with high pseudo Shannon entropy. He then showed how to boost the output to have high pseudo min-entropy by concatenating several independent copies. At this point he used an extractor in NC^0 to turn the output into a pseudorandom string.

However, an extractor in NC^0 needs to have a large seed length (i.e., $\Omega(n)$), thus the NC^0 PRG constructed using this approach only achieves linear stretch. Another issue is that the NC^0 PRG is actually a collection of functions rather than a single function, because the random bits used to sample the bipartite graph is larger than the output length, and is treated as a public index to the collection of functions.

Here, by replacing the extractor with our AC^0 extractor we can achieve a polynomial stretch PRG (based on appropriate assumptions as in [2]), although now the PRG is in AC^0 instead of NC^0. In addition, we can get a single PRG instead of a collection of PRG functions, by including the random bits used to sample the bipartite graph as part of the seed. Since in the graph each right vertex only has a constant number d of neighbors, the sampling uses $md \log n$ bits and can be done in AC^0. To ensure that the PRG has a stretch, we take the sampled graph G and apply the *same* graph to several independent copies of n bit input strings. We show that we can still use the method in [2] to argue that this gives a a distribution with high pseudo Shannon entropy. We then use the same method as in [2] to turn it into a distribution with high pseudo min-entropy, and finally we apply our AC^0 extractor. This way we ensure that the $md \log n$ bits used to sample the graph G are "absorbed" by the stretch of the PRG, and thus we get a standard PRG instead of a collection of PRG functions.

For PRGs for space bounded computation, we simply adapt the PRG by Nisan and Zuckerman [32], which stretches $O(S)$ random bits to any $\mathsf{poly}(S)$ bits that fool space S computation. We now replace the seeded extractor used there by our AC^0 extractor. Notice that the Nisan-Zuckerman PRG simply applies the seeded extractor iteratively for a constant number of times, so the whole construction is still in AC^0.

7 Open problems

Our work leaves many natural open problems. First, in terms of the seed length and output length, our AC^0 extractor is only optimal when $k = \Omega(n)$. Is it possible to simultaneously achieve optimal seed length and output length when $k = n/\mathsf{poly}(\log n)$? Second, can we construct good AC^0 extractors for other classes of sources, such as independent sources and affine sources?

Turning to strong extractor families with small locality, again the parameters of our constructions do not match the parameters of optimal seeded extractors. In particular, our seed length is still $O(k)$ when the min-entropy k is small. Can we reduce the seed length further? We note that using our analysis together with the IW-generator/extractor, one can get something meaningful (i.e., a strong extractor family with a relatively short seed and small locality) even when $k = n^\alpha$ for some $\alpha > 1/2$. But it's unclear how to get below this entropy.

For pseudorandom generators in AC^0, there are also many interesting open problems left. For example, can we construct better cryptographic PRGs, or use weaker computational assumptions? In particular, it would be nice to construct a cryptographic PRG with polynomial stretch based on the one-wayness of a random local function with $m = (1 + \alpha)n$ instead of $m = n^{1+\alpha}$ as in our current construction. For space bounded computation, is it possible to match the exponentially small error of the Nisan-Zukerman PRG? Taking one step further, is it possible to construct PRGs in AC^0 for space bounded computation, with stretch matching the PRGs of Nisan [30] and Impagliazzo-Nisan-Wigderson [20]?

References

1. M. Ajtai and N. Linial. The influence of large coalitions. *Combinatorica*, 13(2):129–145, 1993.
2. Benny Applebaum. Pseudorandom generators with long stretch and low locality from random local one-way functions. *SIAM Journal on Computing*, 42(5):2008–2037, 2013.
3. Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography in nc 0. *SIAM Journal on Computing*, 2006.
4. Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. On pseudorandom generators with linear stretch in nc0. *Computational Complexity*, 17(1):38–69, 2008.
5. L. Babai, L. Fortnow, N. Nisan, and A. Wigderson. Bpp has subexponential simulation unless Exptime has publishable proofs. *Computational Complexity*, 3:307–318, 1993.
6. Andrej Bogdanov and Siyao Guo. Sparse extractor families for all the entropy. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*, pages 553–560. ACM, 2013.
7. Ran Canetti, Yevgeniy Dodis, Shai Halevi, Eyal Kushilevitz, and Amit Sahai. Exposure-resilient functions and all-or-nothing transforms. In Bart Preneel, editor, *Advances in Cryptology — EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 453–469. Springer-Verlag, 2000.
8. J. L. Carter and M. N. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences*, 18:143–154, 1979.
9. Eshan Chattopadhyay and David Zuckerman. Explicit two-source extractors and resilient functions. In *Proceedings of the 48th Annual ACM Symposium on Theory of Computing*, 2016.
10. Kuan Cheng and Xin Li. Randomness extraction in ac0 and with small locality. *arXiv preprint arXiv:1602.01530*, 2016.
11. Anindya De and Luca Trevisan. Extractors using hardness amplification. In *RANDOM 2009, 13th International Workshop on Randomization and Approximation Techniques in Computer Science*, 2009.
12. Stefan Dziembowski and Ueli Maurer. Tight security proofs for the bounded-storage model. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 341–350. ACM, 2002.
13. Ariel Gabizon, Ran Raz, and Ronen Shaltiel. Deterministic extractors for bit-fixing sources by obtaining an independent seed. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 394–403, 2004.
14. Oded Goldreich. Candidate one-way functions based on expander graphs. In *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation*, pages 76–87. Springer, 2011.
15. Oded Goldreich, Emanuele Viola, and Avi Wigderson. On randomness extraction in ac0. In *30th Conference on Computational Complexity (CCC 2015)*, volume 33, pages 601–668. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015.
16. M. Grohe and N. Schweikardt. Lower bounds for sorting with few random accesses to external memory. In *Symposium on Principles of Database Systems (PODS)*, pages 238–249, 2005.
17. Venkatesan Guruswami, Christopher Umans, and Salil Vadhan. Unbalanced expanders and randomness extractors from Parvaresh-Vardy codes. *Journal of the ACM*, 56(4), 2009.
18. J. Håstad. Almost optimal lower bounds for small depth circuits. In S. Micali, editor, *Randomness and Computation*, volume 5, pages 143–170. JAI Press, 1989.
19. Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. Construction of a pseudo-random generator from any one-way function. *SIAM Journal on Computing*, 28:12–24, 1993.

20 R. Impagliazzo, N. Nisan, and A. Wigderson. Pseudorandomness for network algorithms. In *Proceedings of the 26th Annual ACM Symposium on Theory of Computing*, pages 356–364, 1994.

21 R. Impagliazzo and A. Wigderson. P = BPP if E requires exponential circuits: Derandomizing the XOR lemma. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, pages 220–229, 1997.

22 Russell Impagliazzo. Hard-core distributions for somewhat hard problems. In *Foundations of Computer Science, 1995. Proceedings., 36th Annual Symposium on*, pages 538–545. IEEE, 1995.

23 Russell Impagliazzo and Moni Naor. Efficient cryptographic schemes provably as secure as subset sum. *Journal of cryptology*, 9(4):199–216, 1996.

24 Jesse Kamp and David Zuckerman. Deterministic extractors for bit-fixing sources and exposure-resilient cryptography. *SIAM Journal on Computing*, 36(5):1231–1247, 2007.

25 Xin Li. Design extractors, non-malleable condensers and privacy amplification. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing*, pages 837–854, 2012.

26 Xin Li. Improved two-source extractors, and affine extractors for polylogarithmic entropy. In *Proceedings of the 57th Annual IEEE Symposium on Foundations of Computer Science*, 2016.

27 Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, fourier transform, and learnability. *Journal of the ACM*, pages 607–620, 1993.

28 Raghu Meka. Explicit resilient functions matching Ajtai-Linial. *CoRR*, abs/1509.00092, 2015. `arXiv:1509.00092`.

29 Elchanan Mossel, Amir Shpilka, and Luca Trevisan. On ε-biased generators in nc0. *Random Structures & Algorithms*, 29(1):56–81, 2006.

30 Noam Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12:449–461, 1992.

31 Noam Nisan and Avi Wigderson. Hardness vs randomness. *Journal of Computer and System Sciences*, 49(2):149–167, 1994.

32 Noam Nisan and David Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–52, 1996.

33 Ryan O'Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.

34 Periklis A Papakonstantinou, David P Woodruff, and Guang Yang. True randomness from big data. *Scientific reports*, 6, 2016.

35 R. Raz, O. Reingold, and S. Vadhan. Error reduction for extractors. In *Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science*, pages 191–201, 1999.

36 Luca Trevisan. Extractors and pseudorandom generators. *Journal of the ACM*, pages 860–879, 2001.

37 S. Vadhan. On constructing locally computable extractors and cryptosystems in the bounded storage model. In *Advances in Cryptology — CRYPTO '03, 23rd Annual International Cryptology Conference, Proceedings*, 2003.

38 Emanuele Viola. The complexity of constructing pseudorandom generators from hard functions. *computational complexity*, 13(3-4):147–188, 2005.

39 Emanuele Viola. On constructing parallel pseudorandom generators from one-way functions. In *Computational Complexity, 2005. Proceedings. Twentieth Annual IEEE Conference on*, pages 183–197. IEEE, 2005.

40 D. Zuckerman. Randomness-optimal oblivious sampling. *Random Structures and Algorithms*, 11:345–367, 1997.

A Lower bounds for Error parameters of AC^0 extractors

Our negative results about the error of AC^0 extractors follow by a simple application of Fourier analysis and the well known spectrum concentration theorem of AC^0 functions [27].

▶ **Theorem 16** (LMN Theorem [27] [33]). *Let $f : \{-1,1\}^n \to \{-1,1\}$ be computable by AC^0 circuits of size $s > 1$ and depth $\mathtt{dth}$. Let $\epsilon \in (0, 1/2]$. There exists $t = O(\log(s/\epsilon))^{\mathtt{dth}-1} \cdot \log(1/\epsilon)$ s.t.*

$$\sum_{S\subseteq[n],|S|>t} \hat{f}_S^2 \leq \epsilon.$$

Now we prove Theorem 2.

Proof of Theorem 2. Without loss of generality, let $m = 1$. Let's transform the function space of Ext to $\{-1,1\}^{n+d} \to \{-1,1\}$, achieving function f. Let $\epsilon_0 = 1/2$. By Theorem 16, there exists $t = O(\log(s/\epsilon_0))^{\mathtt{dth}-1} \cdot \log(1/\epsilon_0) = O(\log s)^{\mathtt{dth}-1}$ s.t.

$$\sum_{S\subseteq[n+d],|S|\leq t} \hat{f}_S^2 > 1 - \epsilon_0 = 1/2.$$

Fix an $S = S_1 \cup \{i + n \mid i \in S_2\}$ with $|S| \leq t$, where $S_1 \subseteq [n], S_2 \subseteq \{1, 2, \ldots, d\}$. We know that

$$\hat{f}_S = \langle f, \chi_S \rangle = 1 - 2\Pr_u[f(u) \neq \chi_S(u)]$$

where u is uniformly drawn from $\{-1,1\}^{n+d}$.

For $a \in \{-1,1\}$, let X_a be the uniform distribution over $\{-1,1\}^n$ conditioned on $\prod_{i\in S_1} X_i = a$. Also for $b \in \{-1,1\}$, let R_b be the uniform distribution over $\{-1,1\}^d$ conditioned on $\prod_{i\in S_2} R_i = b$. So $\chi_S(x \circ r) = ab$ for $x \in \mathsf{supp}(X_a), r \in \mathsf{supp}(R_b)$. For special situations, saying $S_1 = \emptyset$ (or $S_2 = \emptyset$), let X_a (or R_b) be uniform.

As Ext is a strong (k, ϵ)-extractor, R_b only blows up the error by 2. Also note that by definition, X_a has entropy $n - 1$. So

$$\mathsf{dist}(f(X_a \circ R_b), U) \leq 2\epsilon,$$

where U is uniform over $\{-1,1\}$.

So

$$\forall a, b \in \{-1,1\}, |\Pr[f(X_a \circ R_b) \neq ab] - 1/2| \leq 2\epsilon.$$

Thus

$$\begin{aligned} |\Pr_u[f(u) \neq \chi_S(u)] - 1/2| &= |\sum_{a\in\{-1,1\}} \sum_{b\in\{-1,1\}} \frac{1}{4}(\Pr[f(X_a \circ R_b) \neq ab] - 1/2)| \\ &\leq \sum_{a\in\{-1,1\}} \sum_{b\in\{-1,1\}} \frac{1}{4}|\Pr[f(X_a \circ R_b) \neq ab] - 1/2| \\ &\leq 2\epsilon. \end{aligned} \tag{1}$$

Hence

$$|\hat{f}_S| = |1 - 2\Pr_u[f(u) \neq \chi_S(u)]| = 2|\Pr_u[f(u) \neq \chi_S(u)] - 1/2| \leq 4\epsilon.$$

As a result,

$$1/2 \leq \sum_{S\subseteq[n+d],|S|\leq t} \hat{f}_S^2 \leq \sum_{i=0}^{t} \binom{n+d}{i} (4\epsilon)^2.$$

So

$$\epsilon \geq \sqrt{\frac{1}{32\sum_{i=0}^{t}\binom{n+d}{i}}}.$$

As $\sum_{i=0}^{t}\binom{n+d}{i} \leq (\frac{e(n+d)}{t})^t = 2^{O(t\log(n+d))} = 2^{O(\log s)^{\mathtt{dth}-1}\log(n+d)}$, $\epsilon = 2^{-O(\log s)^{\mathtt{dth}-1}\log(n+d)}$.

The second assertion follows from a similar argument which is deferred to the full version [10]. ◀

B The basic construction of extractors in AC^0

Our basic construction is based on the general idea of I-W generator [21]. In [36], Trevisan showed that I-W generator is an extractor if we regard the string x drawn from the input (n,k)-source X as the truth table of a function f_x s.t. $f_x(\langle i\rangle), i \in [n]$ outputs the ith bit of x.

The construction of I-W generator involves a process of hardness amplifications from a worst-case hard function to an average-case hard function. There are mainly 3 amplification steps. Viola [38] summarizes these results in details, and we review them again. The first step is established by Babai et al. [5], which is an amplification from worst-case hardness to mildly average-case hardness.

▶ **Definition 17.** A boolean function $f : \{0,1\}^l \to \{0,1\}$ is δ-hard on uniform distributions for circuit size g, if for any circuit C with at most g gates ($\mathsf{size}(C) \leq g$), we have $\Pr_{x\leftarrow U}[C(x) = f(x)] < 1-\delta$.

▶ **Lemma 18** ([5]). *If there is a boolean function $f : \{0,1\}^l \to \{0,1\}$ which is 0-hard for circuit size $g = 2^{\Omega(l)}$ then there is a boolean function $f' : \{0,1\}^{\Theta(l)} \to \{0,1\}$ that is $1/\mathsf{poly}(l)$-hard for circuit size $g' = 2^{\Omega(l)}$.*

The second step is an amplification from mildly average-case hardness to constant average-case hardness, established by Impagliazzo [22].

▶ **Lemma 19** ([22]).

1. *If there is a boolean function $f : \{0,1\}^l \to \{0,1\}$ that is δ-hard for circuit size g where $\delta < 1/(16l)$, then there is a boolean function $f' : \{0,1\}^{3l} \to \{0,1\}$ that is $0.05\delta l$-hard for circuit size $g' = \delta^{O(1)}l^{-O(1)}g$.*

   $$f'(s,r) = \langle s, f(a_1)\circ f(a_2)\circ\cdots\circ f(a_l)\rangle$$

   Here $|s| = l$, $|r| = 2l$ and $|a_i| = l, \forall i \in [l]$. Regarding r as a uniform random string, $a_1,\ldots,a_l$ are generated as pairwise independent random strings from the seed r.
2. *If there is a boolean function $f : \{0,1\}^l \to \{0,1\}$ that is δ-hard for circuit size g where $\delta < 1$ is a constant, then there is a boolean function $f' : \{0,1\}^{3l} \to \{0,1\}$ that is $1/2 - O(l^{-2/3})$-hard for circuit size $g' = l^{-O(1)}g$, where*

   $$f'(s,r) = \langle s, f(a_1)\circ f(a_2)\circ\cdots\circ f(a_l)\rangle.$$

   Here $|s| = l$, $|r| = 2l$ and $|a_i| = l, \forall i \in [l]$. Regarding r as a uniform random string, $a_1,\ldots,a_l$ are generated as pairwise independent random strings from the seed r.

The first part of this lemma can be applied for a constant number of times to get a function having constant average-case hardness. After that the second part is usually applied for only once to get a function with constant average-case hardness such that the constant is large enough (at least 1/3).

The third step is an amplification from constant average-case hardness to even stronger average-case hardness, developed by Impagliazzo and Widgerson [21]. Their construction uses the following Nisan-Widgerson Generator [31] which is widely used in hardness amplification.

▶ **Definition 20** ((n,m,k,l)-design and Nisan-Widgerson Generator [31]). A system of sets $S_1, S_2, \ldots, S_m \subseteq [n]$ is an (n,m,k,l)-design, if $\forall i \in [m], |S_i| = l$ and $\forall i,j \in [m], i \neq j, |S_i \cap S_j| \leq k$.

Let $\mathcal{S} = \{S_1, S_2, \ldots, S_m\}$ be an (n,m,k,l) design and $f : \{0,1\}^l \to \{0,1\}$ be a boolean function. The Nisan-Widgerson Generator is defined as $\mathsf{NW}_{f,\mathcal{S}}(u) = f(u|_{S_1}) \circ f(u|_{S_2}) \circ \cdots \circ f(u|_{S_m})$. Here $u|_{S_i} = u_{i_1} \circ u_{i_2} \circ \cdots \circ u_{i_m}$ assuming $S_i = \{i_1, \ldots, i_m\}$.

Nisan and Widgeson [31] showed that the (n,m,k,l)-design can be constructed efficiently.

▶ **Lemma 21** (Implicit in [31]). *For any $\alpha \in (0,1)$, for any large enough $l \in \mathbb{N}$, for any $m < \exp\{\frac{\alpha l}{4}\}$, there exists an $(n,m,\alpha l,l)$-design where $n = \lfloor \frac{10l}{\alpha} \rfloor$. This design can be computed in time polynomial of 2^n.*

The following is the third step of hardness amplification.

▶ **Lemma 22** (Implicit in [21]). *For any $\gamma \in (0, 1/30)$, if there is a boolean function $f : \{0,1\}^l \to \{0,1\}$ that is $1/3$-hard for circuit size $g = 2^{\gamma l}$, then there is a boolean function $f' : \{0,1\}^{l'=\Theta(l)} \to \{0,1\}$ that is $(1/2-\epsilon)$-hard for circuit size $g' = \Theta(g^{1/4}\epsilon^2 l^{-2})$ where $\epsilon \geq (500l)^{1/3} g^{-1/12}$.*

$$f'(a, s, v_1, w) = \langle s, f(a|_{S_1} \oplus v_1) \circ f(a|_{S_2} \oplus v_2) \circ \cdots f(a|_{S_l} \oplus v_l) \rangle$$

Here $(S_1, \ldots, S_l)$ is an $(|a|, l, \gamma l/4, l)$-design where $|a| = \lfloor \frac{40l}{\gamma} \rfloor$. The vectors $v_1, \ldots, v_l$ are obtained by a random walk on an expander graph, starting at v_1 and walking according to w where $|v_1| = l, |w| = \Theta(l)$. The length of s is l. So $l' = |a| + |s| + |v_1| + |w| = \Theta(l)$.

Our basic construction is an adjustment of the IW-Extractor.

▶ **Construction 23.** *For any $c_2 \in \mathbb{N}^+$ such that $c_2 \geq 2$ and any $k = \Theta(n/\log^{c_2-2} n)$, let X be an (n,k)-source . We construct a strong $(k, 2\epsilon)$ extractor $\mathsf{Ext}_0 : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ where $\epsilon = 1/n^\beta$, $\beta = 1/600$, $d = O(\log n)$, $m = k^{\Theta(1)}$. Let U be the uniform distribution of length d.*

1. *Draw x from X and u from U. Let $f_1 : \{0,1\}^{l_1} \to \{0,1\}$ be a boolean function such that $\forall i \in [2^{l_1}]$, $f_1(\langle i \rangle) = x_i$ where $l_1 = \log n$.*
2. *Run amplification step of Lemma 19 part 1 for c_2 times and run amplification step of Lemma 19 part 2 once to get function $f_2 : \{0,1\}^{l_2} \to \{0,1\}$ from f_1 where $l_2 = 3^{c_2+1} l_1 = \Theta(\log n)$.*
3. *Run amplification step Lemma 22 to get function $f_3 : \{0,1\}^{l_3} \to \{0,1\}$ from f_2 where $l_3 = \Theta(\log n)$.*
4. *Construct function Ext_0 such that $\mathsf{Ext}_0(x,u) = \mathsf{NW}_{f_3,\mathcal{S}}(u)$.*

Here $\mathcal{S} = \{S_1, S_2, \ldots, S_m\}$ is a $(d, m, \theta l_3, l_3)$-design with $\theta = l_1/(900 l_3)$, $d = \lfloor 10 l_3/\theta \rfloor$, $m = \lfloor 2^{\frac{\theta l_3}{4}} \rfloor = \lfloor n^{\frac{1}{3600}} \rfloor$.

▶ **Lemma 24.** *In Construction 23, Ext_0 is a strong $(k, 2\epsilon)$ extractor.*

The proof follows from the "Bad Set" argument given by Trevisan [36]. In Trevisan [36] the argument is not explicit for strong extractors. Here our argument is explicit for proving that our construction gives a strong extractor.

Proof. We will prove that for every (n,k)-source X and for every $A : \{0,1\}^{d+m} \to \{0,1\}$ the following holds.

$$|\Pr[A(U_s \circ \mathsf{Ext}_0(X,U_s)) = 1] - \Pr[A(U) = 1]| \leq 2\epsilon$$

Here U_s is the uniform distribution over $\{0,1\}^d$ and U is the uniform distribution over $\{0,1\}^{d+m}$.

For every flat (n,k)-source X, and for every (fixed) function A, let's focus on a set $B \subseteq \{0,1\}^n$ such that $\forall x \in \mathsf{supp}(X)$, if $x \in B$, then

$$|\Pr[A(U_s \circ \mathsf{Ext}_0(x,U_s)) = 1] - \Pr[A(U) = 1]| > \epsilon.$$

According to Nisan and Widgerson [31], we have the following lemma.

▶ **Lemma 25** (Implicit in [31] [36]). *If there exists an A-gate such that*

$$|\Pr[A(U_s \circ \mathsf{Ext}_0(x,U_s)) = 1] - \Pr[A(U) = 1]| > \epsilon,$$

then there is a circuit C_3 of size $O(2^{\theta l_3} m)$, using A-gates, that can compute f_3 correctly for $1/2 + \epsilon/m$ fraction of inputs.

Here A-gate is a special gate that can compute the function A.

By Lemma 25, there is a circuit C_3 of size $O(m2^{\theta l_3}) = O(2^{\frac{5\theta l_3}{4}}) = O(n^{1/720})$, using A-gates, that can compute f_3 correctly for $1/2 + \epsilon/m \geq 1/2 + 1/n^{1/400}$ fraction of inputs.

By Lemma 22, there is a circuit C_2, with A-gates, of size at most $\Theta(n^{\frac{1}{30}})$ which can compute f_2 correctly for at least $2/3$ fraction of inputs.

According to Lemma 19 and our settings, there is a circuit C_1, with A-gates, of size $n^{\frac{1}{30}}\mathsf{poly}\log n$ which can compute f_1 correctly for at least $1 - 1/(c_1 \log^{c_2} n)$ fraction of inputs for some constant $c_1 > 0$.

Next we give an upper bound on the size of B. $\forall x \in B$, assume we have a circuit of size $S = n^{1/30}\mathsf{poly}(\log n)$, using A-gates, that can compute at least $1 - 1/(c_1 \log^{c_2} n)$ fraction of bits of x. The total number of circuits, with A-gates, of size S is at most $2^{\Theta(mS \log S)} = 2^{n^{1/15}\mathsf{poly}(\log n)}$, as A is fixed and has fan-in $m + d = O(m)$. Each one of them corresponds to at most $\sum_{i=0}^{n/(c_1 \log^{c_2} n)} \binom{n}{i} \leq (e \cdot c_1 \log^{c_2} n)^{n/(c_1 \log^{c_2} n)} = 2^{O(n/\log^{c_2-1} n)}$ number of x. So

$$|B| \leq 2^{n^{1/15}\mathsf{poly}(\log n)} 2^{O(n/\log^{c_2-1} n)} = 2^{O(n/(\log^{c_2-1} n)}.$$

As X is an (n,k)-source with $k = \Theta(n/\log^{c_2-2} n)$,

$$\Pr[X \in B] \leq |B| \cdot 2^{-k} \leq \epsilon.$$

Then we know,

$$\begin{aligned}
&|\Pr[A(U_s \circ \mathsf{Ext}(X,U_s)) = 1] - \Pr[A(U) = 1]| \\
=&\sum_{x \in B} \Pr[X = x] |\Pr[A(U_s \circ \mathsf{Ext}(x,U_s)) = 1] - \Pr[A(U) = 1]| \\
&+ \sum_{x \notin B} \Pr[X = x] |\Pr[A(U_s \circ \mathsf{Ext}(x,U_s)) = 1] - \Pr[A(U) = 1]| \\
\leq& 2\epsilon. \qquad (2)
\end{aligned}$$

◀

▶ **Lemma 26.** *The seed length of construction 23 is* $O(\log n)$.

Proof. We know that $l_1 = \log n, l_2 = 3^{c_2+1}l_1 = \Theta(\log n), l_3 = \Theta(\log n)$. Also $\mathcal{S}$ is a $(\lceil 10l_3/c \rceil = \Theta(l_3), m, cl_3, l_3)$-design. So $d = \lfloor 10l_3/c \rfloor = \Theta(l_3) = \Theta(\log n)$. ◀

▶ **Lemma 27.** *The function* Ext_0 *in Construction 23 is in* AC^0. *The circuit depth is* $c_2 + 5$. *The locality is* $\Theta(\log^{c_2+2} n) = \mathsf{poly}(\log n)$.

Please refer to the full version [10] for the proof of this lemma.

According to to Lemma 24, Lemma 26, Lemma 27, we have the following theorem.

▶ **Theorem 28** (The basic construction)**.** *For any* $c \in \mathbb{N}$, *any* $k = \Theta(n/\log^c n)$, *there exists an explicit strong* (k, ϵ)*-extractor* $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *in* AC^0 *of depth* $c + 7$, *where* $\epsilon = n^{-1/600}$, $d = O(\log n)$, $m = \lfloor n^{\frac{1}{3600}} \rfloor$ *and the locality is* $\Theta(\log^{c+4} n) = \mathsf{poly}\log n$.

Boolean Function Analysis on High-Dimensional Expanders

Yotam Dikstein[1]
Weizmann Institute of Science, ISRAEL
yotam.dikstein@weizmann.ac.il

Irit Dinur
Weizmann Institute of Science, ISRAEL
irit.dinur@weizmann.ac.il

Yuval Filmus[2]
Technion – Israel Institute of Technology, ISRAEL
yuvalfi@cs.technion.ac.il

Prahladh Harsha[3]
Tata Institute of Fundamental Research, INDIA
prahladh@tifr.res.in

Abstract

We initiate the study of Boolean function analysis on high-dimensional expanders. We describe an analog of the Fourier expansion and of the Fourier levels on simplicial complexes, and generalize the FKN theorem to high-dimensional expanders.

Our results demonstrate that a high-dimensional expanding complex X can sometimes serve as a sparse model for the Boolean slice or hypercube, and quite possibly additional results from Boolean function analysis can be carried over to this sparse model. Therefore, this model can be viewed as a derandomization of the Boolean slice, containing $|X(k)| = O(n)$ points in comparison to $\binom{n}{k+1}$ points in the $(k+1)$-slice (which consists of all n-bit strings with exactly $k+1$ ones).

2012 ACM Subject Classification Theory of computation → Randomness, geometry and discrete structures

Keywords and phrases high dimensional expanders, Boolean function analysis, sparse model

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.38

Related Version A full version of the paper appears in the `arXiv:1804.08155` [3].

1 Introduction

Boolean function analysis is an essential tool in theory of computation. Traditionally, it studies functions on the Boolean cube $\{-1,1\}^n$. Recently, the scope of Boolean function analysis has extended further, encompassing groups [10, 9, 29, 11], association schemes [27, 13, 14, 16, 15, 6, 23], error-correcting codes [1], and quantum Boolean functions [26]. Boolean function analysis on extended domains has led to progress in learning theory [27] and on the unique games conjecture [7, 22, 2, 23].

[1] The research of the first and second authors was supported in part by Irit Dinur's ERC-CoG grant 772839.
[2] Taub Fellow – supported by the Taub Foundations. The research was funded by ISF grant 1337/16.
[3] Research supported in part by UGC–ISF grant. Part of the work was done when the author was visiting the Weizmann Institute of Science.

Another essential tool in theory of computation is expander graphs. Recently, high-dimensional expanders (HDXs), originally constructed by Lubotzky et al. [25, 24], have been introduced into theory of computation, with applications to property testing [5] and lattices [20]. Just as expander graphs are sparse models of the complete graph, so are high-dimensional expanders sparse models of the complete *hypergraph*, and hence can be used both for derandomization and to improve constructions of objects such as PCPs. *The goal of this work is to connect these two threads of research, introducing Boolean function analysis on high-dimensional expanders.*

We study Boolean functions on simplicial complexes. A pure d-dimensional simplicial complex X is a set system consisting of an arbitrary collection of sets of size $d+1$ together with all their subsets. The sets in a simplicial complex are called *faces*, and it is standard to denote by $X(i)$ the faces of X whose cardinality is $i+1$. Our simplicial complexes are *weighted* by a probability distribution Π_d on the top-level faces, which induces in a natural way probability distributions Π_i on $X(i)$ for all i. Our main object of study is the space of functions $f\colon X(d) \to \mathbb{R}$ (which are often called $\mathbb{R}$-cochains), and in particular, Boolean functions $f\colon X(d) \to \{0,1\}$.

While much of our work applies to arbitrary complexes, our goal is to study complexes which are *high-dimensional expanders*. There are several different non-equivalent ways to define high-dimensional expanders, generalizing various properties of expander graphs. The notion most appropriate to us is the two-sided[4] spectral definition of high-dimensional expanders, due to Dinur and Kaufman [5], who also show how to construct such complexes using the Ramanujan complexes of Lubotzky, Samuels and Vishne [25, 24].

There are several works on random walks on high dimensional expanders which naturally lead to analyzing both real-valued and Boolean valued functions on $X(d)$, for example see [20, 5, 21]. The most related work is by Kaufman and Oppenheim [21], which we discuss in Section 4.3.

Every function on the Boolean cube $\{-1,1\}^n$ has a unique representation as a multilinear polynomial, known as its *Fourier expansion*. The multilinear monomials can be partitioned into "levels" according to their degree, and this corresponds to an orthogonal decomposition of a function into a sum of its homogeneous parts, $f = \sum_{i=0}^{\deg f} f^{=i}$, a decomposition which is a basic concept in Boolean function analysis.

These concepts have known counterparts for the *complete complex*, which consists of all subsets of $[n]$ size at most $d+1$, where $d+1 \le n/2$. The facets (top-level faces) of this complex comprise the *slice* (as it is known for computer scientists) or the *Johnson scheme* (as it is known for coding theorists), whose spectral theory has been elucidated by Dunkl [8]. For $|t| \le d+1$, let $y_t(s) = 1$ iff $t \subseteq s$ (these are the analogs of monomials). Every function on the complete complex has a unique representation as a linear combination of monomials $\sum_t \tilde{f}(t) y_t$ (of various degrees) satisfying the *harmonicity* condition: for all $i \le d$ and all $t \in X(i)$,

$$\sum_{a \in [n] \setminus t} \tilde{f}(t \cup \{a\}) = 0.$$

(If we identify y_t with the product $\prod_{i \in t} x_i$ of "variables" x_i, then harmonicity of a multilinear polynomial P translates to the condition $\sum_{i=1}^n \frac{\partial P}{\partial x_i} = 0$.) As in the case of the Boolean

[4] A related and slightly weaker notion of one-sided spectral expansion has already appeared earlier in [19, 12].

cube, this unique representation allows us to orthogonally decompose a function into its homogeneous parts (corresponding to the contribution of monomials y_t with fixed $|t|$), which plays the same essential part in the complete complex as its counterpart does in the Boolean cube. Moreover, this unique representation allows extending a function from the "slice" to the Boolean cube (which can be viewed as a superset of the "slice"), thus implying further results such as invariance principle [15, 16].

We generalize these concepts for complexes satisfying a technical condition we call *properness*, which is satisfied by both the complete complex as well as high-dimensional expanders. We show that the results on unique decomposition for the complete complex hold for arbitrary proper complexes, with a generalized definition of harmonicity which incorporates the distributions Π_i. In contrast to the case of the complete complex (and the Boolean cube), the homogeneous parts are only approximately orthogonal.

The homogeneous components in our decomposition are "approximate eigenfunctions" of the Laplacian, and this allows us to derive an approximate identity relating the total influence (defined through the Laplacian) to the norms of the components in our decomposition, in complete analogy to the same identity in the Boolean cube (expressing the total influence in terms of the Fourier expansion). All of this is summarized in Theorem 4.1.

As a demonstration of the power of this setup, we generalize the fundamental result of Friedgut, Kalai, and Naor [17] on Boolean functions almost of degree 1. We view this as a first step toward developing a full-fledged theory of Boolean functions on high-dimensional expanders. An easy exercise shows that a Boolean degree 1 function on the Boolean cube is a *dictator*, that is, depends on at most one coordinate; we call this the *exact FKN theorem*. The FKN theorem states that a Boolean function on the Boolean cube which is *close* to a degree 1 function is in fact close to a dictator, where closeness is measured in L_2.

The exact FKN theorem holds for the complete complex as well. Recently, the second author [13] extended the FKN result to the complete complex. Surprisingly, the class of approximating functions has to be extended beyond just dictators.

We prove an exact FKN theorem for arbitrary proper complexes, and an FKN theorem for high-dimensional expanders. In contrast to the complete complex, Boolean degree 1 functions on arbitrary complexes correspond to *independent sets* rather than just single points, and this makes the proof of the exact FKN theorem non-trivial. Our proof of the FKN theorem for high-dimensional expanders is very different from existing proofs. It follows the same general plan as our recent work on the biased Kindler–Safra theorem [4]. The idea is to view a high-dimensional expander as a convex combination of small sub-complexes, each of which is isomorphic to the complete k-dimensional complex on $O(k)$ vertices. We can apply the known FKN theorem separately on each of these, and deduce that our function is approximately well structured on each sub-complex. Finally, we apply the agreement theorem of Dinur and Kaufman [5] to show that the same thing is true on a global level.

1.1 Results

Given a positive integer n and $d < n/2$, let X be the d-dimensional simplex consisting of all subsets of $\{1, \ldots, n\}$ of size at most $d+1$. For $\ell \leq d$, let $X(\ell)$ denote the set of subsets of size exactly $\ell+1$. In the following, we will view any function $f\colon X(\ell) \to \mathbb{R}$ as a function mapping n-bit strings $(y_1, \ldots, y_n) \in \{0,1\}^n$ satisfying $y_1 + \cdots + y_n = \ell + 1$ to the reals. A classical result (see for example [16]) states that every such function has a unique representation $f = f_{-1} + \cdots + f_\ell$ satisfying the following properties:

- f_i is a homogeneous multilinear polynomial of degree $i+1$.

- Each f_i is in the kernel of $\sum_{j=1}^{n} \frac{\partial}{\partial y_j}$.
- $\|f\|^2 = \|f_{-1}\|^2 + \cdots + \|f_\ell\|^2$.
- $\langle DUf, f\rangle = \sum_{i=-1}^{\ell} \left(1 - \frac{i+1}{d-\ell}\right) \frac{\ell-i+1}{\ell+2} \|f_i\|^2$.

Here $DU\colon \mathbb{R}^{X(\ell)} \to \mathbb{R}^{X(\ell)}$ is the (upper) Laplacian, given by the formula

$$(DUf)(s) = \mathop{\mathbb{E}}_{j \notin s} \mathop{\mathbb{E}}_{k \in s \cup \{j\}} f\big((s \cup \{j\}) \setminus \{k\}\big).$$

Our first result is an analogous decomposition for functions on any high-dimensional expander (not necessarily the complete complex):

▶ **Theorem 1.1** (Decomposition theorem for functions on HDX). *Let X be a d-dimensional expander. Every function $f\colon X(\ell) \to \mathbb{R}$ for $\ell \le d$, can be written uniquely as $f = f_{-1} + \cdots + f_\ell$ such that:*

- *f_i is a linear combination of the functions $y_s(t) = [t \supseteq s]$ for $s \in X(i)$.*
- *Interpreted as a function on $X(i)$, f_i lies in the kernel of the "Down" operator.*
- *$\|f\|^2 \approx \|f_{-1}\|^2 + \cdots + \|f_\ell\|^2$.*
- *If $\ell < k$ then $\langle DUf, f\rangle \approx \sum_{i=-1}^{\ell} \frac{\ell-i+1}{\ell+2} \|f_i\|^2$.*

The "Down" operator and the Laplacian are defined formally in Section 2. They differ from their counterparts in the complete complex by taking into account the measure Π_d.

Equipped with this decomposition, we prove the following exact and approximate FKN theorem on high dimensional expanders.

▶ **Theorem 1.2** (exact FKN theorem on HDX). *Let X be a d-dimensional expander. If $f\colon X(d) \to \{0,1\}$ has degree 1, then f is the indicator of either intersecting or not intersecting an independent set of X.*

▶ **Theorem 1.3** (FKN theorem on HDX). *Let X be a d-dimensional expander. If $F\colon X(d) \to \{0,1\}$ is ε-close (in L_2^2) to a degree-1 function then there exists a degree-1 function g on $X(d)$ such that $\Pr[F \neq g] = O(\varepsilon)$.*

Paper organization

We describe our general setup in Section 2. We describe the property of properness and its implications – a unique representation theorem and decomposition of functions into homogeneous parts – in Section 3, discussing the issue of orthogonality of the homogeneous parts in Section 4. Theorem 4.1 summarizes these results. We discuss high-dimensional expanders from our perspective in Section 5. We prove our exact FKN theorem in Section 6, and our FKN theorem in Section 7.

Theorem 1.1 is a combination of Theorem 3.2 (first two items), Theorem 4.1 (other two items), Lemma 4.7 (calculation of the coefficients in the last item) and Theorem 5.1 (showing that high-dimensional expanders satisfy the prerequisites of the preceding results). Theorem 1.2 is a restatement of Theorem 6.1. Theorem 1.3 is a restatement of Theorem 7.2.

2 Basic setup

A d-dimensional complex X is a non-empty collection of sets of size at most $d+1$. We call a set of size $i+1$ an *i-dimensional face* (or i-face for short), and denote the collection of all i-faces by $X(i)$. A d-dimensional complex X is *pure* if every i-face is a subset of some d-face. We will only be interested in pure complexes.

Let X be a pure d-dimensional complex. Given a probability distribution Π_d on its top-dimensional faces $X(d)$, for each $i < d$ we define a distribution Π_i on the i-faces using the following experiment: choose a top-dimensional face according to Π_d, and remove $d-i$ points at random. We can couple all of these distributions to a random vector $\vec{\Pi} = (\Pi_d, \ldots, \Pi_{-1})$ of which the individual distributions are marginals.

Let $C^i = \{f \colon X(i) \to \mathbb{R}\}$ be the space of functions on $X(i)$. It is convenient to define $X(-1) := \{\emptyset\}$, and we also let $C^{-1} = \mathbb{R}$. We turn C^i to an inner product space by defining $\langle f, g\rangle := \mathbb{E}_{\Pi_i}[fg]$ and the associated norm $\|f\|^2 := \mathbb{E}_{\Pi_d}[f^2]$.

For $-1 \le i < d$, we define the Up operator $U_i \colon C^i \to C^{i+1}$ as follows:[5]

$$U_i g(s) := \frac{1}{i+2} \sum_{x \in s} g(s \setminus \{x\}) = \mathop{\mathbb{E}}_{t \subset s}[g(t)] \,,$$

where t is obtained from s by removing a random element. Note that if $s \sim \Pi_{i+1}$ then $t \sim \Pi_i$.

Similarly, we define the Down operator $D_{i+1} \colon C^{i+1} \to C^i$ for $-1 \le i < d$ as follows:

$$D_{i+1} f(t) := \frac{1}{(i+2) \cdot \Pi_i(t)} \sum_{x \notin t \colon t \cup \{x\} \in X(i+1)} \Pi_{i+1}(t \cup \{x\}) \cdot f(t \cup \{x\}) = \mathop{\mathbb{E}}_{s \supset t}[f(s)] \,,$$

where s is obtained from t by conditioning the vector $\vec{\Pi}$ on $\Pi_i = t$ and taking the $(i+1)$th component.

The operators U_i, D_{i+1} are adjoint to each other. Indeed, if $f \in C^{i+1}$ and $g \in C^i$ then

$$\langle g, D_{i+1} f\rangle = \mathop{\mathbb{E}}_{(t,s) \sim (\Pi_i, \Pi_{i+1})}[g(t) f(s)] = \langle U_i g, f\rangle \,.$$

When the domain is understood, we will use U, D instead of U_i, D_{i+1}. This will be especially useful when considering powers of U, D. For example, if $f \colon X(i) \to \mathbb{R}$ then

$$U^t f \equiv U_{i+t-1} \ldots U_{i+1} U_i f.$$

The function y_s is the indicator function of containing s. Our definition of the Up operator guarantees the correctness of the following lemma.

▶ **Lemma 2.1.** *Let $s \in X(i)$. We can think of y_s as a function in C^j for all $j \ge i$. Using this convention, $U_j y_s = (1 - \frac{i+1}{j+2}) y_s$.*

Proof. Direct calculation shows that

$$(U_j y_s)(t) = \frac{1}{j+2} \sum_{x \in t} y_s(t \setminus \{x\}) = \frac{|t| - |s|}{j+2} y_s(t) \,,$$

and so $U_j y_s = (1 - \frac{i+1}{j+2}) y_s$. ◀

For $0 \le i \le k$, the space of harmonic functions on $X(i)$ is

$$H^i := \ker D_i = \{f \in C^i : D_i f = 0\} \,.$$

We also define $H^{-1} = C^{-1}$. We are interested in decomposing C^k, so let us define for each $-1 \le i \le k$,

$$V^i := U^{k-i} H^i = \{U^{k-i} f \ : \ f \in H^i\} \,.$$

We can describe V^i, a sub-class of functions of c^k, in more concrete terms.

[5] The Up and Down operators differ from the boundary and coboundary operators of algebraic topology, which operate on linear combinations of *oriented* faces.

▶ **Lemma 2.2.** *Every function $h \in V^i$ has a unique representation of the form*

$$h = \sum_{s \in X(i)} \tilde{h}(s) y_s \ ,$$

where the coefficients $\tilde{h}(s)$ satisfy the following harmonicity *condition: for all $t \in X(i-1)$,*

$$\sum_{s \supset t} \Pi_i(s) \tilde{h}(s) = 0 \ .$$

Proof. First, let us show that every function of the form given above is in V^i. Lemma 2.1 shows that a function h is of the form $\sum_{s \in X(i)} \tilde{h}(s) y_s$ where $\tilde{h}$ satisfies the harmonicity condition if and only if $h = U^{k-i} r$, where $r \in C^i$ is of similar form $r = \sum_{s \in X(i)} \tilde{r}(s) y_s$, where $\tilde{r}$ satisfies the harmonicity condition. In fact, it is easy to see that $r = \tilde{r}$, and so the fact that $r \in H^i$ follows directly from the definition of the Down operator.

We have shown above that if $h = U^{k-i} r$ then the coefficients $\tilde{h}$ are a constant multiple of the values of r, hence this representation is also unique. ◀

3 Decomposition of the space C^k and a convenient basis

Our decomposition theorem relies on a crucial property of complexes, *properness*.

▶ **Definition 3.1.** A k-dimensional complex is *proper* if it is pure and $\ker U_i$ is trivial for $-1 \leq i \leq k-1$.

The complete k-dimensional complex on n points is proper iff $k+1 \leq n/2$. A pure one-dimensional complex (i.e., a graph) is proper iff it is not bipartite. Unfortunately, we are not aware of a similar characterization for higher dimensions. However, in Section 5 we show that high-dimensional expanders are proper.

We can now state our decomposition theorem.

▶ **Theorem 3.2.** *If X is a proper k-dimensional complex then we have the following decomposition of C^k:*

$$C^k = V^k + V^{k-1} + \cdots + V^{-1} \ .$$

In other words, for every function $f \in C^k$ there is a unique choice of $h_i \in H^i$ such that the functions $f_i = U^{k-i} h_i$ satisfy $f = f_{-1} + f_0 + \ldots + f_k$.

Proof. We first prove by induction that every function $f \in C^\ell$ has a representation $f = \sum_{i=-1}^{\ell} U^{\ell-i} h_i$, where $h_i \in H^i$. This trivially holds when $\ell = -1$. Suppose now that the claim holds for some $\ell < k$, and let $f \in C^{\ell+1}$. Since $D^{\ell+1} \colon C^{\ell+1} \to C^\ell$ is a linear operator, we have $C^{\ell+1} = \ker D_{\ell+1} + \operatorname{im} D^*_{\ell+1} = \ker D_{\ell+1} + \operatorname{im} U_\ell$, and therefore we can write $f = h_{\ell+1} + Ug$, where $h_{\ell+1} \in H^{\ell+1}$ and $g \in C^\ell$. Applying induction, we get that $g = \sum_{i=-1}^{\ell} U^{\ell-i} h_i$, where $h_i \in H^i$. Substituting this in $f = h_{\ell+1} + Ug$ completes the proof.

It remains to show that the representation is unique. Since $\ker U_{i-1} = \ker D_i^*$ is trivial, $\dim H^i = \dim C^i - \dim C^{i-1}$ for $i \geq 0$. This shows that $\sum_{i=-1}^{k} \dim H^i = \dim C^k$. Therefore the operator $\varphi \colon H^{-1} \times \cdots \times H^k \to C^k$ given by $\varphi(h_{-1}, \ldots, h_k) = \sum_{i=-1}^{k} U^{k-i} h_i$ is not only surjective but also injective. In other words, the representation of f is unique. ◀

▶ **Corollary 3.3.** *If X is a proper k-dimensional complex then every function $f \in C^k$ has a unique representation of the form*

$$f = \sum_{s \in X} \tilde{f}(s) y_s \ ,$$

where the coefficients $\tilde{f}(s)$ *satisfy the following* harmonicity *conditions: for all* $0 \le i \le k$ *and all* $t \in X(i-1)$*:*

$$\sum_{\substack{s \in X(i) \\ s \supset t}} \Pi_i(s) \tilde{f}(s) = 0 \ .$$

Proof. Follows directly from Lemma 2.2. ◀

We can now define the degree of a function.

▶ **Definition 3.4.** The *degree* of a function f is the maximal cardinality of a face s such that $\tilde{f}(s) \neq 0$ in the unique decomposition given by Corollary 3.3.

Thus a function has degree d if its decomposition only involves faces whose dimension is less than d. The following lemma shows that the functions y_s, for all $(d-1)$-dimensional faces s, span the space of all functions of degree at most d.

▶ **Lemma 3.5.** *If* X *is a proper* k*-dimensional complex then the space of functions on* $X(k)$ *of degree at most* $d+1$ *has the functions* $\{y_s : s \in X(d)\}$ *as a basis.*

Proof. The space of functions on $X(k)$ of degree at most $d+1$ is spanned, by definition, by the functions y_t for $t \in X(-1) \cup X(0) \cup \cdots \cup X(d)$. This space has dimension $\sum_{i=-1}^{d} \dim H^i$. Since X is proper, $\dim H^i = \dim C^i - \dim C^{i-1}$ for $i > 0$, and so $\sum_{i=1}^{d} \dim H^i = \dim C^d = |X(d)|$.

Given the above in order to complete the proof it suffices to show that every y_t, $t \in X(i), i \le d$, can be written as a linear combination of y_s for $s \in X(d)$. This will show that $\{y_s : s \in X(d)\}$ spans the space of functions of degree at most $d+1$. Since this set contains $|X(d)|$ functions, it forms a basis.

Recall that $y_t(r) = 1_{r \supset t}$, where $r \in X(k)$. If r contains t then it contains exactly $\binom{k+1-|t|}{d+1-|t|}$ many d-faces containing r, and so

$$y_t = \frac{1}{\binom{k+1-|t|}{d+1-|t|}} \sum_{\substack{s \supset t \\ s \in X(d)}} y_s.$$

This completes the proof. ◀

We call f_i the "level i" part of f, and denote the weight of f above level i by

$$wt_{>i}(f) := \sum_{j>i} \|f_j\|_2^2.$$

4 Orthogonality of decomposition

When X is the complete k-dimensional complex, the decomposition in Theorem 3.2 is orthogonal. For a general complex, this no longer need be the case. However, under certain conditions, the decomposition is *almost* orthogonal, as we show in this subsection, proving the following result:

▶ **Theorem 4.1.** *Let* $\vec{r}, \vec{\delta}$ *be vectors such that pointwise* $\vec{r} > \vec{0}$ *and* $\vec{\delta} < \vec{1}$*.*
Let X *be a proper* k*-dimensional complex, and define*

$$\gamma := \max_{0 \le i \le k-1} \|D_{i+1}U_i - (1-\delta_i)U_{i-1}D_i - r_i\|.$$

For every function f *on* C^ℓ *for* $\ell \le k$*, the decomposition* $f = f_{-1} + \cdots + f_\ell$ *of Theorem 3.2 satisfies the following properties:*

- *For $i \neq j$, $|\langle f_i, f_j\rangle| = O(\gamma)\|f_i\|\|f_j\|$.*
- *$\|f\|^2 = (1\pm O(\gamma))(\|f_{-1}\|^2+\cdots+\|f_\ell\|^2)$, and for all i, $\|f\|^2 = (1\pm O(\gamma))(\|f_{\le i}\|^2+\|f_{>i}\|^2)$.*
- *If $\ell < k$ then $\langle DUf, f\rangle = (1 \pm O(\gamma)) \sum_{i=-1}^{\ell} \lambda_i \|f_i\|^2$, where $\vec{\lambda}$ depends only on $\ell, \vec{r}, \vec{\delta}$.*

In other words, the decomposition of Theorem 3.2 is almost orthogonal, and its parts are almost eigenfunctions of the Laplacian operator DU. The (unnormalized) Laplacian operator is used in classical Boolean function analysis to define both the total influence $\mathrm{Inf}[f] = \langle DUf, f\rangle$ and the noise operator (in the semigroup formulation, $T_t = e^{-tDU}$).

As we show in Section 4.4 and Section 5, for both the complete complex and high-dimensional expanders we can take $r_i = \delta_i = \frac{1}{i+2}$, and given $\ell < k$, we get $\lambda_i = 1 - \frac{i+1}{\ell+2}$.

Since the first two properties for arbitrary ℓ follow from the case $\ell = k$ by truncating the complex, we concentrate below on proving this special case.

4.1 Sequentially differential posets

Let us first discuss *sequentially differential posets* [30, 31], using the example of the *unnormalized complete complex* whose top-level faces consists of all subsets of $[n]$ of size $k+1$, weighted according to the *counting measure*. The top-level faces of this complex form the "slice" $\binom{[n]}{k+1}$ (as it is known by computer scientists) of the *Johnson scheme* $J(n, k+1)$ (as it is known by coding theorists). The Up operator $\tilde{U}_i \colon C^i \to C^{i+1}$ is given by

$$\tilde{U}_i g(s) := \sum_{x \in s} g(s \setminus \{x\}),$$

and the Down operator $\tilde{D}_{i+1} \colon C^{i+1} \to C^i$ is given by

$$\tilde{D}_{i+1} f(t) := \sum_{y \notin t} f(t \cup \{y\}).$$

A simple calculation shows that

$$\tilde{D}_{i+1}\tilde{U}_i g(t) = \sum_{y\notin t} \sum_{x \in t\cup\{y\}} g(t \cup \{y\} \setminus \{x\}) = (n-i-1)g(t) + \sum_{r \colon |r\setminus t| = |t\setminus r| = 1} g(r),$$

$$\tilde{U}_{i-1}\tilde{D}_i g(t) = \sum_{x\in t} \sum_{y \notin t\setminus\{x\}} g(t \setminus \{x\} \cup \{y\}) = (i+1)g(t) + \sum_{r \colon |r\setminus t| = |t\setminus r| = 1} g(r).$$

Comparing the two expressions, we see that

$$\tilde{D}_{i+1}\tilde{U}_i - \tilde{U}_{i-1}\tilde{D}_i = r_i I, \text{ where } r_i = n - 2(i+1),$$

where I is the identity operator. Posets satisfying this property, for an arbitrary vector $\vec{r} = r_0, \ldots, r_{k-1}$, are known as *sequentially differential posets*. An important example is the *Grassmann lattice* of all subspaces of a finite-dimensional vector space over a finite field.

In the setup considered in this paper, the top-level faces are weighted by a distribution rather than an arbitrary measure. We therefore consider the *(normalized) complete complex*, in which the top-level faces are weighted by the uniform distribution. It is easy to check that all distributions Π_i are uniform, and therefore

$$U_i g(s) = \frac{1}{i+2} \sum_{x \in s} g(s \setminus \{x\}),$$

$$D_{i+1} f(t) = \frac{1}{n-i-1} \sum_{y \notin t} f(t \cup \{y\}).$$

As before, we can calculate

$$\begin{aligned} D_{i+1}U_i g(t) &= \frac{1}{(n-i-1)(i+2)} \sum_{y \notin t} \sum_{x \in t \cup \{y\}} g(t \cup \{y\} \setminus \{x\}) \\ &= \frac{1}{i+2} g(t) + \frac{1}{(n-i-1)(i+2)} \sum_{r\colon |r \setminus t| = |t \setminus r| = 1} g(r), \\ U_{i-1}D_i g(t) &= \frac{1}{(i+1)(n-i)} \sum_{x \in t} \sum_{y \notin t \setminus \{x\}} g(t \setminus \{x\} \cup \{y\}) \\ &= \frac{1}{n-i} g(t) + \frac{1}{(i+1)(n-i)} \sum_{r\colon |r \setminus t| = |t \setminus r| = 1} g(r). \end{aligned}$$

Comparing the two expressions, we see that

$$D_{i+1}U_i - (1-\delta_i)U_{i-1}D_i = r_i I, \text{ where } r_i = \delta_i = \frac{1}{i+2} - \frac{i+1}{i+2} \cdot \frac{1}{n-i-1}. \tag{1}$$

Since U_{i-1} and D_i are adjoint, $U_{i-1}D_i$ is positive semidefinite. When $2(i+1) < n$, the constant r_i is positive, and so $D_{i+1}U_i$ is positive definite, which in particular implies that U_i has trivial kernel. This shows that the complete complex is proper when $k < n/2$.

4.2 Almost sequentially differential posets

High-dimensional expanders do not satisfy an identity of the form (1). However, they satisfy an *approximate* version for of this identity, as we show in Section 5. The classical theory of sequentially differential posets shows that the decomposition of Theorem 3.2 is orthogonal. We will now show that an approximate version of (1) suffices for approximate orthogonality.

Given a positive vector $\vec{r}$, a vector $\vec{\delta}$ with coordinates less than 1, and a parameter γ, let us say that a k-dimensional complex is $(\vec{r}, \vec{\delta}, \gamma)$-*almost sequentially differential*, or $(\vec{r}, \vec{\delta}, \gamma)$-ASD for short, if for $0 \leq i \leq k-1$,

$$\|D_{i+1}U_i - (1-\delta_i)U_{i-1}D_i - r_i\| \leq \gamma,$$

where the norm is the spectral norm.

We start by showing that in such a complex, any two distinct parts in the decomposition of Theorem 3.2 are approximately orthogonal, in some sense.

▶ **Lemma 4.2.** *Suppose that X is a proper k-dimensional complex which is $(\vec{r}, \vec{\delta}, \gamma)$-ASD, and let $f \in C^k$ have the decomposition $f = f_{-1} + \cdots + f_k$ for $f_i = U^{k-i}h_i$, as in Theorem 3.2. For $i \neq j$,*

$$\langle f_i, f_j \rangle = O(\gamma)\|h_i\|\|h_j\|.$$

where the hidden constant depends only on $k, \vec{r}, \vec{\delta}$ but not on $n = |X(0)|$.

Proof. Recall that $h_i \in H^i = \ker D_i$. Given this, it is easy to see that f_k is orthogonal to f_{k-1}, indeed $\langle f_k, f_{k-1} \rangle = \langle h_k, Uh_{k-1} \rangle = \langle Dh_k, h_{k-1} \rangle = 0$ because $Dh_k = 0$. To warm up let us first prove the claim for $f_{k-1} = Uh_{k-1}$ and $f_{k-2} = U^2 h_{k-2}$. In this case $\langle f_{k-1}, f_{k-2} \rangle = \langle Uh_{k-1}, U^2 h_{k-2} \rangle = \langle D^2 U h_{k-1}, h_{k-2} \rangle$. If we could replace $D^2 U h_{k-1}$ by $(1-\delta_{k-1})DUDh_{k-1} + r_{k-1}Dh_{k-1} = 0$ as in a sequentially differential poset, we would be done. However, this is not necessarily true. We instead use the property of being $(\vec{r}, \vec{\delta}, \gamma)$-ASD and replace $DDUh_{k-1}$ with $(1-\delta_{k-1})DUDh_{k-1} + r_{k-1}Dh_{k-1} = 0$, incurring an error of $O(\gamma)$ and completing the argument in this case.

Now move to general i, j and suppose without loss of generality that $j < i$, and so $k-j > k-i$. We have $\langle f_i, f_j\rangle = \langle D^{k-j}U^{k-i}h_i, h_j\rangle$. Let us denote $E_i = D_{i+1}U_i - U_{i-1}D_i - r_i$. We expand $D^{k-j}U^{k-i}$ into a sum of terms, using the following algorithm. We start with the sum containing one term, $D^{k-j}U^{k-i}$. At each step, we pick a term not containing E and not of the form U^aD^b, and isolate one of the occurrences of DU, say the term is $c\alpha DU\beta$ (here c is a real number and α, β are products of operators). We replace this term with the terms $c(1-\delta_d)\alpha UD\beta$, $cr_d\alpha\beta$ (for the appropriate d), and $c\alpha E\beta$ (this corresponds to the identity $D_{i+1}U_i = (1-\delta_i)U_{i-1}D_i + r_i + E_i$). This process clearly terminates eventually, with an expression that depends on $k, \vec{r}, \vec{\delta}$ but not on n.

Since $k - j > k - i$, all terms either contain E or end with D. The terms ending with D vanish since $h_i \in H^i$. All other terms are of the form $c\langle \alpha E\beta h_i, h_j\rangle = c\langle E\beta h_i, \alpha^* h_j\rangle$, where β, α^* are products of Ds and Us. Since D and U are contractions, we can estimate

$$|\langle E\beta h_i, \alpha^* h_j\rangle| \le \|E\beta h_i\|\|\alpha^* h_j\| \le \gamma\|h_i\|\|h_j\|.$$

The lemma immediately follows. ◀

The preceding lemma gives an error estimate in terms of the norms $\|h_i\|$. The following lemma will enable us to express the error in terms of the norms $\|f_i\|$.

▶ **Lemma 4.3.** *Suppose that X is a proper k-dimensional complex which is $(\vec{r}, \vec{\delta}, \gamma)$-ASD, and let $f \in C^k$ have the decomposition $f = f_{-1} + \cdots + f_k$ for $f_i = U^{k-i}h_i$, as in Theorem 3.2. For every i there exists a constant ρ_i, depending only on $\vec{r}, \vec{\delta}$, such that*

$$\|f_i\| = (1 \pm O(\gamma))\rho_i\|h_i\|,$$

where the hidden constant depends only on $k, \vec{r}, \vec{\delta}$.

Proof. The argument is very similar to that of Lemma 4.2. This time we are computing $\|f_i\|^2 = \langle D^{k-i}U^{k-i}h_i, h_i\rangle$. Executing the same algorithm as before, we will be left with many terms of the form $\langle h_i, h_i\rangle$, with various coefficients depending only on $\vec{r}, \vec{\delta}$. The end result will be that $\|f_i\|^2 = \rho_i^2\|h_i\|^2 \pm O(\gamma)\|h_i\|^2$ for some ρ_i (note that the coefficient is positive since $\vec{r}$ is positive and all entries of $\vec{\delta}$ are less than 1). The lemma follows. ◀

Combining both lemmata, we obtain the following corollary.

▶ **Corollary 4.4.** *Suppose that X is a proper k-dimensional complex which is $(\vec{r}, \vec{\delta}, \gamma)$-ASD, and let $f \in C^k$ have the decomposition $f = f_{-1} + \cdots + f_k$, as in Theorem 3.2. If γ is small enough (as a function of $k, \vec{r}, \vec{\delta}$) then for $i \ne j$,*

$$\langle f_i, f_j\rangle = O(\gamma)\|f_i\|\|f_j\|,$$

where the hidden constant depends only on $k, \vec{r}, \vec{\delta}$.

As a consequence, we obtain an *approximate L2 mass formula*:

▶ **Corollary 4.5.** *Under the conditions of Corollary 4.4, for every $i \le j$ we have*

$$\|f_i + \cdots + f_j\|^2 = (1 \pm O(\gamma))(\|f_i\|^2 + \cdots + \|f_j\|^2),$$

where the hidden constant depends only on $k, \vec{r}, \vec{\delta}$.

In particular,

$$\|f\|^2 = (1 \pm O(\gamma))(wt_{\le i}(f) + wt_{>i}(f)) = (1 \pm O(\gamma))(\|f_{\le i}\|^2 + \|f_{>i}\|^2).$$

Proof. Expanding $\|f_i + \cdots + f_j\|^2$, we obtain

$$\|f_i + \cdots + f_j\|^2 - \|f_i\|^2 - \cdots - \|f_j\|^2 = 2 \sum_{i \le a < b \le j} \langle f_a, f_b \rangle = O(\gamma) \sum_{i \le a < b \le j} \|f_a\| \|f_b\| \le$$
$$O(\gamma) \left(\|f_i\| + \cdots + \|f_j\|\right)^2 \le O(\gamma)(\|f_i\|^2 + \cdots + \|f_j\|^2),$$

swallowing a factor of k in the last inequality. ◀

4.3 Laplacian

The (upper) Laplacian is the operator DU, used to define a random walk on a specific level $\ell < k$ of the complex (the lower Laplacian UD defines another random walk). In the complete complex, indeed in arbitrary complexes satisfying (1), the functions in the decomposition $f = f_{-1} + \cdots + f_\ell$ are eigenfunctions of the Laplacian. The same holds approximately for almost sequentially differential posets, using arguments very similar to the foregoing.

An analog of Lemma 4.2 shows that for $i \neq j$,

$$\langle Uf_i, Uf_j \rangle = O(\gamma) \|h_i\| \|h_j\|,$$

which as in Corollary 4.4 shows that

$$\langle Uf_i, Uf_j \rangle = O(\gamma) \|f_i\| \|f_j\|.$$

The argument of Corollary 4.5 shows that

$$\langle DUf, f \rangle = \|Uf\|^2 = (1 \pm O(\gamma))(\|Uf_{-1}\|^2 + \cdots + \|Uf_\ell\|^2).$$

Conversely, an analog of Lemma 4.3 shows that there are positive constants λ_i, depending only on $\vec{r}, \vec{\delta}$, such that for each i,

$$\|Uf_i\|^2 = (1 \pm O(\gamma))\lambda_i \rho_i^2 \|h_i\|^2 = (1 \pm O(\gamma))\lambda_i \|f_i\|^2,$$

using Lemma 4.3. Putting everything together, we get the following result:

▶ **Lemma 4.6.** *Suppose that X is a proper k-dimensional complex which is $(\vec{r}, \vec{\delta}, \gamma)$-ASD, let $\ell < k$, and let $f \in C^\ell$ have the decomposition $f = f_{-1} + \cdots + f_\ell$, as in Theorem 3.2. For every i there exists a constant λ_i, depending only on $\vec{r}, \vec{\delta}$, such that*

$$\langle DUf, f \rangle = (1 \pm O(\gamma)) \sum_{i=-1}^{\ell} \lambda_i \|f_i\|^2,$$

where the hidden constant depends only on $k, \vec{r}, \vec{\delta}$.

This result is analogous to [21, Theorem 6.2], in which a similar decomposition is obtained. However, whereas our decomposition is to functions $f_{-1}, \ldots, f_\ell$ in C^ℓ, the decomposition of [21] is analogous to our functions $h_{-1}, \ldots, h_\ell$, which live in different spaces.

4.4 Eigenvalues of the Laplacian

Section 4.1 shows that the complete d-dimensional complex is $(\vec{r}, \vec{\delta}, \gamma)$-ASD, where $r_i = \delta_i = \frac{1}{i+2}$ and $\gamma = O(\frac{1}{n-d-1})$, the same parameters as high-dimensional expanders, as we show in Section 5. This allows us to compute the eigenvalues $\vec{\lambda}$ of the Laplacian in Lemma 4.6 for both the complete complex and high-dimensional expanders.

The classical theory of the "slice" (see for example [14, 16]) shows that V^i is spanned by functions of the form $\prod_{j=1}^{i+1}(y_{a_j} - y_{b_j})$, where the $2i$ indices a_j, b_j are distinct. (Recall that in the decomposition of f, the component f_i belongs to V^i.)

▶ **Lemma 4.7.** *Let $\varphi_i \in C^\ell$ be the function*

$$\varphi_i = (y_1 - y_2)(y_3 - y_4)\cdots(y_{2i+1} - y_{2i+2}).$$

On the complete complex,

$$DU\varphi_i = \left(1 - \frac{i+1}{n-\ell-1}\right)\left(1 - \frac{i+1}{\ell+2}\right)\varphi_i.$$

Proof. Let $s \in X(\ell)$, and recall that

$$DU\varphi_i(s) = \mathop{\mathbb{E}}_{a\notin s}\mathop{\mathbb{E}}_{b\in s\cup\{a\}} \varphi_i(s\cup\{a\}\setminus\{b\}).$$

For brevity, we use $r_{a,b} := s\cup\{a\}\setminus\{b\}$.

We consider several cases. Suppose first that $2j-1, 2j \in s$ for some j, so that $\varphi_i(s) = 0$. If $b \neq 2j-1, 2j$ then $\varphi_i(r_{a,b}) = 0$. Conversely, $\varphi_i(r_{a,2j-1}) = -\varphi_i(r_{a,2j})$ for any choice of a. Since $\Pr[b = 2j-1] = \Pr[b = j]$, we see that $DU\varphi_i(s) = 0$.

Suppose next that $2j-1, 2j \notin s$ for some j, so that again $\varphi_i(s) = 0$. If $a \notin 2j-1, 2j$ then $\varphi_i(r_{a,b}) = 0$. Conversely, $\varphi_i(r_{2j-1,b}) = -\varphi_i(r_{2j,b})$ for any $b \in s$, and $\varphi_i(r_{2j-1,2j-1}) = \varphi_i(r_{2j,2j}) = 0$. Once again, this shows that $DU\varphi_i(s) = 0$.

Finally, suppose that $1, 3, \ldots, 2i+1 \in s$ and $2, 4, \ldots, 2i+2 \notin s$, so that $\varphi_i(s) = 1$. If $a = 2j$ for some j then $\varphi_i(r_{a,2j-1}) = -1$, $\varphi_i(r_{a,2j}) = 1$, and $\varphi_i(r_{a,b}) = 0$ for $b \neq 2j-1, 2j$. Thus $\varphi_i(r_{2j,b})$ vanishes in expectation. When $a \neq 2, 4, \ldots, 2i+2$, we have $\varphi_i(r_{a,b}) = 1$ if $b \neq 1, 3, \ldots, 2i+1$, and $\varphi_i(r_{a,b}) = 0$ otherwise. Thus $DU\varphi_i(s)$ is the probability that $a \neq 2, 4, \ldots, 2i+2$ and that $b \neq 1, 3, \ldots, 2i+1$, which is $(1 - \frac{i+1}{n-\ell-1})(1 - \frac{i+1}{\ell+2})$. ◀

When $n \to \infty$, both γ and $\frac{i+1}{n-\ell-1}$ tend to zero. Comparing Lemma 4.6 and Lemma 4.7, we see that the value of λ_i in Lemma 4.6, which doesn't depend on n, is

$$\lambda_i = 1 - \frac{i+1}{\ell+2}.$$

5 High-dimensional expanders

Let X be a d-dimensional complex with an associated probability distribution Π_d on $X(d)$, which induces probability distributions on $X(-1), \ldots, X(d-1)$ as we have described above. For every i-dimensional face $s \in X(i)$ for $i < d-1$, the *link* of s is the weighted graph X_s defined as follows:

- The vertices are points $x \notin s$ such that $s\cup\{x\} \in X(i+1)$ is a face.
- The edges are pairs of points $\{x, y\}$ such that $s\cup\{x, y\} \in X(i+1)$ is a face. (Since the complex is pure, x and y are vertices.)
- The weight of the *directed* edge (x, y) is

$$w_s(x, y) := \frac{1}{2}\Pr_{t\sim\Pi_{i+1}}[t = s\cup\{x, y\} \mid t \supset s].$$

Note that the weights define a probability distribution w_s on the (directed) edges. We denote the marginal of w_s on its first coordinate by

$$w_s(x) := \sum_{y\neq x} w_s(x, y) = \Pr_{t\sim\Pi_{i+1}}[t \supset s\cup\{x\} \mid t \supset s] = \Pr_{(u,v)\sim(\Pi_{i+1},\Pi_i)}[u = s\cup\{x\} \mid v = s].$$

This is also the marginal of w_s on its second coordinate. We turn the space of function on vertices into an inner product space by defining

$$\langle f, g\rangle := \mathop{\mathbb{E}}_{x\sim w_s}[f(x)g(x)].$$

We define an operator A_s on functions on vertices by the matrix $A_s(x,y) := w_s(x,y)/w_s(x)$, which corresponds to the quadratic form

$$\langle f, A_s g\rangle = \sum_{x,y} w_s(x,y)f(x)g(y).$$

By definition, A_s fixes constant functions, and so it is a Markov operator. Since $w_s(x,y) = w_s(y,x)$, it is also self-adjoint with respect to the inner product above. Thus A_s has eigenvalues $\lambda_1 = 1, \lambda_2, \ldots, \lambda_m$, where m is the number of vertices. We define $\lambda(A_s) = \max(|\lambda_2|, |\lambda_m|)$. Orthogonality of eigenspaces guarantees that

$$|\langle f, A_s g\rangle - \mathbb{E}[f]\,\mathbb{E}[g]| \le \lambda(A_s)\|f\|\|g\|.$$

We say that X is a *γ-two-sided high-dimensional expander* (called γ-HD expander in [5]) if *every* link X_s of X satisfies $\lambda(A_S) \le \gamma$.

Let f be a function on $X(i)$, where $i < d-1$. We have

$$\langle DUf, f\rangle = \langle Uf, Uf\rangle = \mathop{\mathbb{E}}_{t\sim\Pi_{i+1}} \mathop{\mathbb{E}}_{x,y\in t}[f(t\setminus\{x\})\cdot f(t\setminus\{y\}].$$

The probability that x is equal to y is exactly $1/(i+2)$, and so

$$\langle DUf, f\rangle = \frac{1}{i+2}\mathop{\mathbb{E}}_{t\sim\Pi_{i+1}}\mathop{\mathbb{E}}_{x\in t}[f(t\setminus\{x\})^2] + \frac{i+1}{i+2}\mathop{\mathbb{E}}_{t\sim\Pi_{i+1}}\mathop{\mathbb{E}}_{x\neq y\in t}[f(t\setminus\{x\})\cdot f(t\setminus\{y\}].$$

If $t\sim\Pi_{i+1}$ and $x\in t$ is chosen at random, then $t\setminus\{x\}\sim\Pi_i$. Therefore the first term is equal to $\frac{1}{i+2}\|f\|^2$. To compute the second term, let $s = t\setminus\{x,y\}$. Since $t\sim\Pi_{i+1}$ and $x\neq y\in t$ are chosen at random, we have $s\sim\Pi_{i-1}$. Given such an s, the probability to get specific (t,x,y) is exactly $w_s(x,y)$ (the factor $1/2$ accounts for the relative order of x,y), and so

$$\langle DUf, f\rangle = \frac{1}{i+2}\|f\|^2 + \frac{i+1}{i+2}\mathop{\mathbb{E}}_{s\sim\Pi_{i-1}}\mathop{\mathbb{E}}_{(x,y)\sim w_s}[f(s\cup\{x\})f(s\cup\{y\})].$$

We now note that

$$\mathop{\mathbb{E}}_{x\sim w_s}[f(s\cup\{x\})] = (Df)(s).$$

Therefore we have

$$\left|\mathop{\mathbb{E}}_{(x,y)\sim w_s}[f(s\cup\{x\})f(s\cup\{y\})] - (Df)(s)^2\right| \le \lambda(A_s)\mathop{\mathbb{E}}_{x\sim w_s}[f(s\cup\{x\})^2].$$

If X is a γ-two-sided high-dimensional expander then $\lambda(A_s)\le\gamma$ for all s, and so (using I for the identity operator)

$$|\langle(DU - \tfrac{1}{i+2}I - \tfrac{i+1}{i+2}UD)f, f\rangle| \le \gamma \mathop{\mathbb{E}}_{s\sim\Pi_{i-1}}\mathop{\mathbb{E}}_{x\sim w_s}[f(s\cup\{x\})^2] = \gamma\|f\|^2.$$

We have proved the following result.

▶ **Theorem 5.1.** *Suppose that X is a d-dimensional γ-two-sided high-dimensional expander. Then its $(d-1)$-skeleton Y (consisting of all faces of X of dimension at most $d-1$) is $(\vec{r},\vec{\delta},\gamma)$-ASD, where $r_i = \delta_i = \frac{1}{i+2}$. Moreover, if $\gamma < \frac{1}{d}$ then Y is proper.*

Proof. The first part follows from the foregoing. For the second part, note that $U_{i-1}D_i$ is positive semidefinite, and so all eigenvalues of $\frac{i+1}{i+2}U_{i-1}D_i + \frac{1}{i+2}$ are at least $\frac{1}{i+2}$. This implies that all eigenvalues $D_{i+1}U_i$ are at least $\frac{1}{i+2} - \gamma > 0$, and in particular U_i has trivial kernel. ◀

Dinur and Kaufman [5] proved that such expanders do exist.

▶ **Theorem 5.2** ([5, Lemma 1.5]). *For every $\lambda > 0$ and every $d \in \mathbb{N}$ there exists an explicit infinite family of bounded degree d-dimensional complexes which are λ-two-sided high-dimensional expanders.*

Theorem 5.1 shows that high-dimensional expanders are almost sequentially differential. In the full version of the paper [3], we show that the converse also holds, given the additional assumption that all links are connected. Our proof uses the local Garland method [18], as substantiated by Oppenheim [28].

6 Boolean degree 1 functions

In this section we characterize all Boolean degree 1 functions in nice complexes.

▶ **Theorem 6.1.** *Suppose that X is a proper k-dimensional complex, where $k \geq 2$. A function $f \in C^k$ is a Boolean degree 1 function if and only if there exists an independent set I such that f is the indicator of intersecting I or of not intersecting I.*

Proof. If f is the indicator of intersecting an independent set I then $f = \sum_{v \in I} y_v$, and so $\deg f \leq 1$. If f is the indicator of not intersecting an independent set I then $f = \sum_{v \in X(0)} y_v/(k+1) - \sum_{v \in I} y_v$, and so again $\deg f \leq 1$.

Suppose now that f is a Boolean degree 1 function. If $|X(0)| \leq 2$ then the theorem clearly holds, so assume that $|X(0)| > 2$. Lemma 3.5 shows that f has a unique representation of the form

$$f = \sum_{v \in X(0)} c_v y_v.$$

Since f is Boolean, it satisfies $f^2 = f$. Note that

$$f^2 = \sum_{\{u,v\} \in X(1)} 2c_u c_v y_{\{u,v\}} + \sum_{v \in X(0)} c_v^2 y_v.$$

Moreover, since every input x to f which contains v contains exactly k other members of $X(0)$, and since $X(1)$ contains all pairs of points from x, we have

$$y_v = \sum_{u \in X(0) \setminus \{v\}} \frac{y_{\{u,v\}}}{k}.$$

This shows that

$$0 = f^2 - f = \sum_{\{u,v\} \in X(1)} 2c_u c_v y_{\{u,v\}} + \frac{1}{k} \sum_{v \in X(0)} (c_v^2 - c_v) \sum_{u \in X(0) \setminus \{v\}} y_{\{u,v\}} = \frac{1}{k} \sum_{\{u,v\} \in X(1)} (2kc_u c_v + c_u^2 - c_u + c_v^2 - c_v) y_{\{u,v\}}.$$

Lemma 3.5 shows that the coefficients of all $y_{\{u,v\}}$ must vanish, that is, for all $u \neq v$ we have

$$2kc_uc_v = c_u(1-c_u) + c_v(1-c_v).$$

Consider now a triple of points u, v, w such that $\{u,v,w\} \in X(2)$, and the corresponding system of equations:

$$\begin{aligned} 2kc_uc_v &= c_u(1-c_u) + c_v(1-c_v), \\ 2kc_uc_w &= c_u(1-c_u) + c_w(1-c_w), \\ 2kc_vc_w &= c_v(1-c_v) + c_w(1-c_w). \end{aligned}$$

Subtracting the second equation from the first, we obtain

$$2kc_u(c_v - c_w) = c_v(1-c_v) - c_w(1-c_w) = (c_v - c_w) - (c_v^2 - c_w^2) = (c_v - c_w)(1 - c_v - c_w).$$

This shows that either $c_v = c_w$ or $2kc_u = 1 - c_v - c_w$.

If $c_u \neq c_v, c_w$ then $2kc_w + c_u + c_v = 2kc_v + c_u + c_w = 1$, which implies that $c_v = c_w$. Thus c_u, c_v, c_w can consist of at most two values. If $c := c_u = c_v = c_w$ then $2kc^2 = 2c(1-c)$, and so $c \in \{0, 1/(k+1)\}$. If $c := c_v = c_w \neq c_u$ then $2c^2 = 2c(1-c)$, and so $c \in \{0, 1/(k+1)\}$ as before. We also have $2kc_uc = c_u(1-c_u) + c(1-c)$. If $c = 0$ then this shows that $c_u(1-c_u) = 0$, and so $c_u = 1$. If $c = 1/(k+1)$ then one can similarly check that $c_u = 1/(k+1) - 1$.

Summarizing, one of the following two cases must happen:

1. Two of c_u, c_v, c_w are equal to 0, and the remaining one is either 0 or 1.
2. Two of c_u, c_v, c_w are equal to $1/(k+1)$, and the remaining one is either $1/(k+1)$ or $1/(k+1) - 1$.

Let us say that a vertex $v \in X(0)$ is of type A if $c_v \in \{0,1\}$, and of type B if $c_v \in \{1/(k+1), 1/(k+1) - 1\}$. Since the complex is pure and at least two-dimensional, every vertex must participate in a triangle (two-dimensional face), and so every vertex is of one of the types. In fact, all vertices must be of the *same* type. Otherwise, there would be a vertex v of type A incident to a vertex w of type B. However, since the complex is pure, $\{v,w\}$ must participate in a triangle, contradicting the classification above.

Suppose first that all vertices are type A, and let $I = \{v : c_v = 1\}$. Note that f indicates that the input face intersects I. Clearly I must be an independent set, since otherwise f would not be Boolean. When all vertices are type B, the function $1 - f = \sum_{v \in X(0)} (1/(k+1) - c_v) y_v$ is of type A, and so f must indicate not intersecting an independent set. ◀

7 FKN-theorem on high dimensional expanders

In this section, we prove an analog of the classical result of Friedgut, Kalai and Naor [17] to high dimensional expanders. The FKN theorem that states that any Boolean function F on the hypercube that is close to a degree-1 function f (not necessarily Boolean) in the L_2^2-sense must agree with some Boolean degree-1 function (which must be a dictator) at most points. This result for the Boolean hypercube can be easily extended to functions on k-slices of the hypercube provided $k = \Theta(n)$.

▶ **Theorem 7.1** (FKN theorem on the slice [13]). *Let $n, k \in \mathbb{Z}_{\geq 0}$ and $\varepsilon \in (0,1)$ such that $n/4 \leq k \leq n/2$. Let $F \colon \binom{[n]}{k} \to \{0,1\}$ be a Boolean function such that $\mathbb{E}[(F-f)^2] < \varepsilon$ for some degree-1 function $f \colon \binom{[n]}{k} \to \{0,1\}$. Then there exists a degree-1 function $g \colon \binom{[n]}{k} \to \mathbb{R}$ such that*

$$\Pr[F \neq g] = O(\varepsilon).$$

Furthermore, $g \in \{0, 1, y_i, 1 - y_i\}$, that is, g is a Boolean dictator (1-junta).

▶ Remark.

1. The function g promised by the theorem satisfies $\mathbb{E}[(g - F)^2] = Pr[g \neq F] = O(\varepsilon)$ and hence, by the L_2^2-triangle inequality we have $\mathbb{E}[(f - g)^2] \leq 2\,\mathbb{E}[(f - F)^2] + 2\,\mathbb{E}[(g - F)^2] = O(\varepsilon)$. This is the way that the FKN theorem is traditionally stated, but we prefer the above formulation as this is the one we are able to generalize to the high-dimensional expander setting.
2. The function 1 can also be written as $\sum_j (1/k) y_j$. The function $1 - y_i$ can also be written as $\sum_{j \neq i} (1/k) y_j + (1/k - 1) y_i$.
3. The result of [13] is quite a bit stronger: for every $k \leq n/2$, it promises the existence of a function $g \colon \binom{[n]}{k} \to \mathbb{R}$, not necessarilly Boolean, such that $\mathbb{E}[(f - g)^2] = O(\varepsilon)$. Moreover, either g or $1 - g$ is of the form $\sum_{i \in S} y_i$ for $|S| \leq \max(1, \sqrt{\varepsilon} \cdot n/k)$. The bound on the size of S ensures that $\Pr[g \in \{0, 1\}] = 1 - O(\varepsilon)$.

Our main theorem is an extension of the above theorem to k-faces of a high-dimensional expander.

▶ **Theorem 7.2** (FKN theorem for high dimensional expanders)**.** *Let X be a d-dimensional λ-two-sided high-dimensional expander, and let $4k^2 < d$ and $\lambda < 1/d$. Let $F \colon X(k) \to \{0, 1\}$ be a function such that $\mathbb{E}[(F - f)^2] < \varepsilon$ for some degree-1 function $f \colon X(k) \to \mathbb{R}$. Then there exists a degree-1 function $g \colon X(k) \to \mathbb{R}$ such that*

$$\Pr[F \neq g] = O_\lambda(\varepsilon).$$

Furthermore, the degree-1 function g can be written as $g(y) = \sum_i d_i y_i$, where $d_i \in \{0, 1, \frac{1}{k+1}, \frac{1}{k+1} - 1\}$.

The high-dimensional analog of the FKN theorem is obtained from the FKN theorem for the slice using the agreement theorem of Dinur and Kaufman [5].

7.1 Agreement theorem for high dimensional expanders

Dinur and Kaufman [5] prove an agreement theorem for high-dimensional expanders. The setup is as follows. For each k-face s we are given a local function $f_s \colon s \to \Sigma$ that assigns values from an alphabet Σ to each point in s. Two local functions $f_s, f_{s'}$ are said to agree if $f_s(v) = f_{s'}(v)$ for all $v \in s \cap s'$. Let $\mathcal{D}_{k,2k}$ be the distribution on pairs (s_1, s_2) obtained by choosing a random $t \sim \Pi_{2k}$ and then independently choosing two k-faces $s_1, s_2 \subset t$. The theorem says that if a random pair pair of faces $(s, s') \sim \mathcal{D}_{k,2k}$ satisfies with high probability that f_s agrees with $f_{s'}$ on their intersection, then there must be a global function $g \colon X(0) \to \Sigma$ such that almost always $g|_s \equiv f_s$. Formally:

▶ **Theorem 7.3** (Agreement theorem for high-dimensional expanders [5])**.** *Let X be a d-dimensional λ-two-sided high-dimensional expander, and let $k^2 < d$ and $\lambda < 1/d$ and Σ some fixed finite alphabet. Let $\{f_s \colon s \to \Sigma\}_{s \in X(k)}$ be an ensemble of local functions on $X(k)$, i.e. $f_s \in \Sigma^s$ for each $s \in X(k)$. If*

$$\Pr_{(s_1, s_2) \sim \mathcal{D}_{k,2k}} [f_{s_1}|_{s_1 \cap s_2} \equiv f_{s_2}|_{s_1 \cap s_2}] > 1 - \varepsilon$$

then there is a $g \colon X(0) \to \Sigma$ such that

$$\Pr_{s \sim \Pi_k} [f_s \equiv g|_s] \geq 1 - O_\lambda(\varepsilon).$$

While Dinur and Kaufman state the theorem for a binary alphabet, the general version follows in a black box fashion by applying the theorem for binary alphabets $\lceil \log_2 |\Sigma| \rceil$ many times.

7.2 Proof of Theorem 7.2

Let $f, F \in C^k$, where F is a Boolean function and f is a degree-1 function as in the hypothesis of Theorem 7.2. Let

$$\varepsilon_f := \mathop{\mathbb{E}}_s[\mathrm{dist}(f(s), \{0,1\})^2]. \tag{2}$$

We have $\varepsilon_f \leq \varepsilon$. Since f is a degree-1 function, Lemma 3.5 guarantees that there exist $a_i \in \mathbb{R}$ such that $f(y) = \sum_{i \in n} a_i y_i$. Note that here we view the inputs of f as n-bit strings with exactly $k+1$ ones, the rest being zero.

We begin by defining an ensemble of pairs of local functions $\{(f|_t, F|_t)\}_{t \in X(2k)}$, $\{(f|_u, F|_u)\}_{u \in X(4k)}$ which are the restrictions of (f, F) to the $2k$-face t and $4k$-face u. Formally, for any $t \in X(2k)$ and $u \in X(4k)$, consider the restriction of f to t and u defined as follows:

$$f|_t, F|_t \colon \binom{t}{k} \to \mathbb{R}, \qquad f|_t(y) = f(y) = \sum_{i \in t} a_i y_i, \qquad F|_t(y) = F(y),$$

$$f|_u, F|_u \colon \binom{u}{k} \to \mathbb{R}, \qquad f|_u(y) = f(y) = \sum_{i \in u} a_i y_i, \qquad F|_u(y) = F(y).$$

Observe that the $f|_t$'s are degree-1 functions while the $F|_t$'s are Boolean functions (similarly for $f|_u$'s and $F|_u$'s).

Now, define the following quantities

$$\varepsilon_t := \mathop{\mathbb{E}}_{s \colon s \subset t}[\mathrm{dist}(f|_t(s), F|_t(s))^2], \qquad \delta_u := \mathop{\mathbb{E}}_{s \colon s \subset u}[\mathrm{dist}(f|_u(s), F|_u(s))^2].$$

Clearly, $\mathbb{E}_t[\varepsilon_t] = \mathbb{E}_u[\delta_u] = \varepsilon_f \leq \varepsilon$, where ε_f is as in (2).

Let $\alpha_k = \frac{1}{k+1}$. Applying Theorem 7.1 (along with Remark 7) to the functions $(f|_t, F|_t)$ for each $t \in X(2k)$ we have the following claim:

▶ **Claim 7.4.** *For every $t \in X(2k)$, there exists a degree-1 dictator $g_t \colon \binom{t}{k} \to \{0,1\}$ such that*

$$\mathop{\mathbb{E}}_{s \colon s \subset t}[(f|_t - g_t)^2] = O(\varepsilon_t).$$

Furthermore, there exists a function $d_t \colon t \to \{0, 1, \alpha_k, \alpha_k - 1\}$ such that $g_t(y) = \sum_{i \in t} d_t(i) y_i$.

Similarly for each $u \in X(4k)$ we have:

▶ **Claim 7.5.** *For every $u \in X(4k)$, there exists a degree-1 dictator $h_u \colon \binom{u}{k} \to \{0,1\}$ such that*

$$\mathop{\mathbb{E}}_{s \colon s \subset u}[(f|_u - h_u)^2] = O(\delta_u).$$

Furthermore, there exists a function $e_u \colon u \to \{0, 1, \alpha_k, \alpha_k - 1\}$ such that $h_u(y) = \sum_{i \in u} e_u(i) y_i$.

We will now prove that the collection of local functions $\{d_t\}_t$ typically agree with each other. We will then be able to use the agreement theorem Theorem 7.3 to sew these different local functions together to yield a single function $d : X(0) \to \{0, 1, \alpha_k, \alpha_k - 1\}$. This d will determine a global degree-1 function g defined as follows: $g(y) = \sum_{i \in X(0)} d(i) y_i$.

▶ **Claim 7.6.** *There exists a function* $d\colon X(0) \to \{0, 1, \alpha_k, \alpha_k - 1\}$ *such that* $\Pr_t[d_t \equiv d|_t] = 1 - O_\lambda(\varepsilon)$.

Proof. To sew the various d_t together via the agreement theorem, we would like to first bound the probability

$$\Pr_{(t_1,t_2)\sim\mathcal{D}_{2k,4k}}[d_{t_1}|_{t_1\cap t_2} \not\equiv d_{t_2}|_{t_1\cap t_2}] \ .$$

Recall the definition of the distribution $\mathcal{D}_{2k,4k}$: we first pick a set $u \in X(4k)$ according to Π_{4k} and then two $2k$-faces t_1, t_2 of u uniformly and independently. Consider the three functions d_{t_1}, d_{t_2} and e_u. Clearly, if $d_{t_1}|_{t_1\cap t_2} \not\equiv d_{t_2}|_{t_1\cap t_2}$ then one of $e_u|_{t_1} \not\equiv d_{t_1}$ or $e_u|_{t_2} \not\equiv d_{t_2}$ must hold. Thus,

$$\Pr_{(t_1,t_2)\sim\mathcal{D}_{2k,4k}}[d_{t_1}|_{t_1\cap t_2} \not\equiv d_{t_2}|_{t_1\cap t_2}] \le 2 \cdot \Pr_{t,u}[e_u|_t \not\equiv d_t] \ . \tag{3}$$

Thus, it suffices to bound the probability $\Pr_{t,u}[e_u|_t \not\equiv d_t]$ where $u \sim \Pi_{4k}$ and t is a random $2k$-face of u.

For any fixed $t \subset u$, the L_2^2 triangle inequality shows that

$$\mathbb{E}[(h_u|_t - g_t)^2] \le 2\,\mathbb{E}[(h_u|_t - f|_t)^2] + 2\,\mathbb{E}[(f|_t - g_t)^2] = 2\,\mathbb{E}[(h_u|_t - f|_t)^2] + O(\varepsilon_t).$$

Taking expectation over $t \in X(2k)$ conditioned on $t \subset u$, we see that

$$\mathop{\mathbb{E}}_{t\subset u}\mathbb{E}[(h_u|_t - g_t)^2] \le 2\,\mathbb{E}[(h_u - f|_u)^2] + O\left(\mathop{\mathbb{E}}_{t:\ t\subset u}\varepsilon_t\right) = O(\delta_u) + O\left(\mathop{\mathbb{E}}_{t:\ t\subset u}\varepsilon_t\right).$$

Taking expectation over $u \sim \Pi_{4k}$, we now have

$$\mathop{\mathbb{E}}_{u}\mathop{\mathbb{E}}_{t\subset u}\mathbb{E}[(h_u|_t - g_t)^2] = O(\varepsilon).$$

For any fixed $t \subset u$, both $h_u|_t$ and g_t are Boolean degree-1 juntas. Hence either they agree, or $\mathbb{E}[(h_u|_t - g_t)^2] = \Omega(1)$. This shows that $h_u|_t$ disagrees with g_t with probability $O(\varepsilon)$, and so

$$\Pr_{t,u}[e_{u|_t} \not\equiv d_t] = O(\varepsilon).$$

We now return to (3), concluding that

$$\mathop{\mathbb{E}}_{(t_1,t_2)\sim\mathcal{D}_{2k,4k}}[d_{t_1}|_{t_1\cap t_2} \not\equiv d_{t_2}|_{t_1\cap t_2})] = O(\varepsilon).$$

We have thus satisfied the hypothesis of the agreement theorem (Theorem 7.3). Invoking the agreement theorem, we deduce that $\Pr_{t\sim\Pi_{2k}}[d_t \equiv d|_t] = 1 - O_\lambda(\varepsilon)$. ◀

The d's guaranteed by Claim 7.6 naturally correspond to a degree-1 function $g\colon X(k) \to \mathbb{R}$ as follows:

$$g(y) := \sum_{i\in X(0)} d(i)y_i.$$

We now show that this g is mostly Boolean.

▶ **Claim 7.7.** $\Pr_s[g(s) \in \{0, 1\}] = 1 - O_\lambda(\varepsilon)$.

Proof. Since g_t is Boolean valued,

$$\Pr_{s\sim\Pi_k}[g(s)\in\{0,1\}] \geq \Pr_t[g|_t = g_t] = \Pr_t[d|_t \equiv d_t] = 1 - O(\varepsilon).$$ ◀

We now show that g in fact agrees pointwise with F most of the time.

▶ **Claim 7.8.** $\Pr_s[g \neq F] = O(\varepsilon)$.

Proof. Fix any $t \in X(2k)$. We compute $\Pr_{s:\, s\subset t}[F|_t \neq g_t]$ as follows

$$\begin{aligned}\Pr[F|_t \neq g_t] &= \|F|_t - g_t\|^2 && [\text{ Since } F|_t \text{ and } g_t \text{ are both Boolean }]\\ &\leq 2\cdot\|F|_t - f|_t\|^2 + 2\cdot\|f|_t - g_t\|^2\\ &= O(\varepsilon_t) + O(\varepsilon_t) = O(\varepsilon_t).\end{aligned}$$

We can now compute $\Pr_s[F \neq g]$ as follows:

$$\Pr[F\neq g] = \mathop{\mathbb{E}}_t \Pr[F|_t \neq g|_t] \leq \mathop{\mathbb{E}}_t \Pr[F|_t \neq g_t] + \Pr_t[g|_t \neq g_t] = O(\varepsilon) + \Pr_t[d|_t \not\equiv d_t] = O_\lambda(\varepsilon).$$ ◀

This completes the proof of Theorem 7.2.

References

1. Boaz Barak, Parikshit Gopalan, Johan Håstad, Raghu Meka, Prasad Raghavendra, and David Steurer. Making the long code shorter. *SIAM J. Comput.*, 44(5):1287–1324, 2015. (Preliminary version in *53rd FOCS*, 2012). doi:10.1137/130929394.
2. Boaz Barak, Pravesh Kothari, and David Steurer. Small-set expansion in shortcode graph and the 2-to-2 conjecture. (manuscript), 2018.
3. Yotam Dikstein, Irit Dinur, Yuval Filmus, and Prahladh Harsha. Boolean function analysis on high-dimensional expanders. (manuscript), 2018. arXiv:1804.08155.
4. Irit Dinur, Yuval Filmus, and Prahladh Harsha. Low degree almost Boolean functions are sparse juntas. (manuscript), 2017. arXiv:1711.09428.
5. Irit Dinur and Tali Kaufman. High dimensional expanders imply agreement expanders. In *Proc. 58th IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 974–985, 2017. doi:10.1109/FOCS.2017.94.
6. Irit Dinur, Subhash Khot, Guy Kindler, Dor Minzer, and Muli Safra. On non-optimally expanding sets in Grassmann graphs. In *Proc. 50th ACM Symp. on Theory of Computing (STOC)*, 2018. (To appear).
7. Irit Dinur, Subhash Khot, Guy Kindler, Dor Minzer, and Muli Safra. Towards a proof of the 2-to-1 games conjecture? In *Proc. 50th ACM Symp. on Theory of Computing (STOC)*, 2018. (To appear).
8. Charles Dunkl. A Krawtchouk polynomial addition theorem and wreath products of symmetric groups. *Indiana Univ. Math. J.*, 25:335–358, 1976. doi:10.1512/iumj.1976.25.25030.
9. David Ellis, Yuval Filmus, and Ehud Friedgut. A quasi-stability result for dictatorships in S_n. *Combinatorica*, 35(5):573–618, 2015. doi:10.1007/s00493-014-3027-1.
10. David Ellis, Yuval Filmus, and Ehud Friedgut. A stability result for balanced dictatorships in S_n. *Random Structures Algorithms*, 46(3):494–530, 2015. doi:10.1002/rsa.20515.
11. David Ellis, Yuval Filmus, and Ehud Friedgut. Low degree Boolean functions on s_n, with an application to isoperimetry. *Forum of Mathematics, Sigma*, 5:e23, 2017. doi:10.1017/fms.2017.24.
12. Shai Evra and Tali Kaufman. Bounded degree cosystolic expanders of every dimension. In *Proc. 48th ACM Symp. on Theory of Computing (STOC)*, pages 36–48, 2016. doi:10.1145/2897518.2897543.

13 Yuval Filmus. Friedgut-Kalai-Naor theorem for slices of the Boolean cube. *Chic. J. Theoret. Comput. Sci.*, 2016(14), 2016. doi:10.4086/cjtcs.2016.014.

14 Yuval Filmus. An orthogonal basis for functions over a slice of the Boolean hypercube. *Electron. J. Combin.*, 23(1):P1.23, 2016. arXiv:1406.0142.

15 Yuval Filmus, Guy Kindler, Elchanan Mossel, and Karl Wimmer. Invariance principle on the slice. In *Proc. 31st Comput. Complexity Conf.*, volume 50 of *LIPIcs*, pages 15:1–15:10. Schloss Dagstuhl, 2016. doi:10.4230/LIPIcs.CCC.2016.15.

16 Yuval Filmus and Elchanan Mossel. Harmonicity and invariance on slices of the Boolean cube. In *Proc. 31st Comput. Complexity Conf.*, volume 50 of *LIPIcs*, pages 16:1–16:13. Schloss Dagstuhl, 2016. doi:10.4230/LIPIcs.CCC.2016.16.

17 Ehud Friedgut, Gil Kalai, and Assaf Naor. Boolean functions whose Fourier transform is concentrated on the first two levels and neutral social choice. *Adv. Appl. Math.*, 29(3):427–437, 2002. doi:10.1016/S0196-8858(02)00024-6.

18 Howard Garland. p-adic curvature and the cohomology of discrete subgroups of p-adic groups. *Ann. of Math.*, 97(3):375–423, 1973. doi:10.2307/1970829.

19 Tali Kaufman, David Kazhdan, and Alexander Lubotzky. Ramanujan complexes and bounded degree topological expanders. In *Proc. 55th IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 484–493, 2014. doi:10.1109/FOCS.2014.58.

20 Tali Kaufman and David Mass. Good distance lattices from high dimensional expanders. (manuscript), 2018. arXiv:1803.02849.

21 Tali Kaufman and Izhar Oppenheim. High order random walks: Beyond spectral gap. In Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer, editors, *Proc. 20th International Workshop on Randomization and Computation (RANDOM)*, volume 116 of *LIPIcs*. Schloss Dagstuhl, 2018. doi:10.4230/LIPIcs.APPROX/RANDOM.2018.46.

22 Subhash Khot, Dor Minzer, and Muli Safra. On independent sets, 2-to-2 games, and Grassmann graphs. In *Proc. 49th ACM Symp. on Theory of Computing (STOC)*, pages 576–589, 2017. doi:10.1145/3055399.3055432.

23 Subhash Khot, Dor Minzer, and Muli Safra. Pseudorandom sets in Grassmann graph have near-perfect expansion. (manuscript), 2018.

24 Alexander Lubotzky, Beth Samuels, and Uzi Vishne. Explicit constructions of Ramanujan complexes of type $\tilde{A}_d$. *European J. Combin.*, 26(6):965—993, 2005. doi:10.1016/j.ejc.2004.06.007.

25 Alexander Lubotzky, Beth Samuels, and Uzi Vishne. Ramanujan complexes of type $\tilde{A}_d$. *Israel J. Math.*, 149(1):267–299, 2005. doi:10.1007/BF02772543.

26 Ashley Montanaro and Tobias Osborne. Quantum boolean functions. *Chic. J. Theoret. Comput. Sci.*, 2010, 2010. doi:10.4086/cjtcs.2010.001.

27 Ryan O'Donnell and Karl Wimmer. KKL, Kruskal-Katona, and monotone nets. *SIAM J. Comput.*, 42(6):2375–2399, 2013. (Preliminary version in *50th FOCS*, 2009). doi:10.1137/100787325.

28 Izhar Oppenheim. Local spectral expansion approach to high dimensional expanders part I: Descent of spectral gaps. *Discrete Comput. Geom.*, 59(2):293–330, 2018. doi:10.1007/s00454-017-9948-x.

29 Rafael Plaza. Stability for intersecting families in $PGL(2, q)$. *Electron. J. Combin.*, 22(4):P4.41, 2015. URL: http://www.combinatorics.org/ojs/index.php/eljc/article/view/v22i4p41.

30 Richard P. Stanley. Differential posets. *J. Amer. Math. Soc.*, 1(4):919–961, 1988. doi:10.2307/1990995.

31 Richard P. Stanley. Variations on differential posets. In Dennisa Stanton, editor, *Invariant Theory and Tableaux*, volume 19 of *IMA Vol. Math. Appl.*, pages 145–165. Springer, 1990.

Percolation of Lipschitz Surface and Tight Bounds on the Spread of Information Among Mobile Agents

Peter Gracar
Mathematical Institute, University of Cologne, Weyertal 86-90, 50931 Köln, Germany
pgracar@math.uni-koeln.de

Alexandre Stauffer[1]
Department of Mathematical Sciences, University of Bath,
Claverton Down, Bath, BA2 7AY, United Kingdom
a.stauffer@bath.ac.uk

Abstract

We consider the problem of spread of information among mobile agents on the torus. The agents are initially distributed as a Poisson point process on the torus, and move as independent simple random walks. Two agents can share information whenever they are at the same vertex of the torus. We study the so-called flooding time: the amount of time it takes for information to be known by all agents. We establish a tight upper bound on the flooding time, and introduce a technique which we believe can be applicable to analyze other processes involving mobile agents.

2012 ACM Subject Classification Mathematics of computing → Probability and statistics

Keywords and phrases Lipschitz surface, spread of information, flooding time, moving agents

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.39

Related Version `https://arxiv.org/abs/1702.08748`

1 Introduction

We consider the problem of spread of information between mobile agents on a d-dimensional torus of side-length n. We will denote by $N = n^d$ the number of vertices on the torus, and will refer to the agents as *particles*. At time 0, the particles are distributed on the vertices of the torus as a Poisson point process of intensity λ. Then, particles move by performing independent continuous-time simple random walks on the torus; that is, at rate 1 a particle chooses a neighboring vertex uniformly at random and jumps there. It is not difficult to check that this system of particles is in stationarity. Thus, at any given time t, the location of the particles is a Poisson point process of intensity λ on the torus. However, the configuration of particles at time t is not independent of the configuration of particles at time 0, and as we will explain below, it is this dependence that makes this model challenging to analyze.

Assume that at time 0 there is a particle at the origin with a piece of information that has to be distributed to all other particles. Then, any uninformed particle (a particle that does not know the information) receives the information whenever it is at the same vertex as an informed particle (a particle that knows the information). We study the time it takes the information to reach all the particles, which is commonly referred to as the *flooding time*.

[1] Supported by a Marie Curie Career Integration Grant PCIG13-GA-2013-618588 DSRELIS, and an EPSRC Early Career Fellowship.

A big challenge in analyzing this model is due to the heavily dependent structure of the particles. In fact, though particles move independently of one another, dependences do arise over time. For example, if a ball of radius R centered at some vertex x of the torus turns out to have no particles at time 0, then the ball $B(x, R/2)$ of radius $R/2$ centered at x will continue to be empty of particles up to time R^2, with positive probability. This means that the probability that the $(d+1)$-dimensional, space-time cylinder $B(x, R/2) \times [0, R^2]$ has no particle is at least $\exp\{-cR^d\}$ for some constant c. This is just a stretched exponential on the volume of the cylinder, which prevents us from applying classical methods based on comparison with independent percolation [10], since those require exponential decay of correlations. In addition to this, whenever one finds such a ball of radius R empty of particles at time 0, this affects regions of the torus in the vicinity of this ball. In particular, during a time interval of length R^2, the density of particle in the vicinity of the ball will be smaller than the expected density λ. In this work we develop a framework to control such dependences.

When the transmission radius is large (in the sense that information can be transmitted between particles at distance $O(\log^{1/d}(n))$ of each other) or the jump range is large (in the sense that a particle can jump a distance of order $O(\log^{1/d} n)$ in one step), then the dependences can be more easily controlled. These cases where analyzed in [4, 5], where tight bounds on the flooding time (up to constant factors) were established[2]. Having a large transmission radius or jump range helps the analysis because of the following. Tessellate the torus into boxes of side-length $\Theta(\log^{1/d}(n))$, and tessellate time into intervals of constant length. Then, since the system of particles is in stationary and boxes are so large, we can apply a Chernoff bound for Poisson random variables to show that, for any given box and time interval, with probability $1 - n^{-C}$, there is a large enough number of particles inside the box during that time interval (when this happen, call the cell of the tessellation *good*). Then a union bound can be used to show that all cells of the tessellation are good. Then, if the transmission radius is large enough to allow particles from neighboring boxes to exchange information, one can establish a tight bound on the flooding time. If it is the jump range that is large enough, then one can use the fact that, after a time interval of order 1, the configuration of particles inside any given box is close to stationarity. In other words, the system of particles has a small mixing time. This washes away the dependences of the system, and allowed a tight bound (up to constant factors) to be derived.

An important open problem has been to analyze the case where both the transmission radius and the jump range are of order 1, which is our setting here. This was studied in [9], where it was shown that, with high probability, the flooding time is at most $\tilde{\Theta}(n)$, where the notation $\tilde{\Theta}(\cdot)$ means that poly-logarithmic factors are neglected[3]. This bound is tight up to poly-logarithmic factors since, for a transmission radius and jump range of order 1, the flooding time is $\Omega(n)$ in all dimensions. We note that, when neglecting poly-logarithmic factors, one can still work with the above tessellation — of cells of side-length $\Theta(\log^{1/d}(n))$ — for which all cells of the tessellation are good. This is because one can do some suboptimal estimates to allow information to spread inside a cell (thereby losing only a poly-logarithmic factor), and then use the fact that cells are good, and full of particles, to let the information spread from one cell to the next. Getting a bound that is tight up to *constant* factors, on

[2] In fact, [4] studies the flooding time for a larger class of dynamic graphs. However, due to space limitations, we restrict our discussion to results on the specific model of spread of information among random walk particles.

[3] We remark that [9] considers also the case where the number of particles can be of order much smaller than N, and [12, 11] analyze a variant of this model, but these settings are out of the scope of this work.

the other hand, involves a rather delicate issue, since one is forced to consider tessellations of *constant* side-length, which will naturally contain a positive density of bad cells, forcing a more careful control of the dependences of the system.

Turning back to our setting, where particles can jump only across neighboring vertices and information can be transmitted only between particles located at the same vertex, [8] analyzes the process in the whole of $\mathbb{Z}^d$ and shows that the information spreads with positive speed. To prove this, the authors developed a complicated multi-scale framework to control the dependences of the system, where tessellations of different side-lengths were considered and controlled. This multi-scale technique is quite powerful, and has been employed in the mathematics literature to solve other processes with slow decay of correlations [13, 15, 3]. However, this technique is usually very difficult to implement, and has to be tailored to each specific model and question being studied. The goal of our work is to develop a robust and flexible multi-scale framework that can be more easily applied to answer questions involving systems of random walk particles, and we illustrate its usefulness by deriving tight bounds on the flooding time.

1.1 Our results

We start considering a more general setup. Let $\mathbb{T}^d$ be the d-dimensional integer torus of side length n. Let $G = (\mathbb{T}^d, E)$ be the nearest neighbor graph on $\mathbb{T}^d$. Let $\{\mu_{x,y}\}_{(x,y)\in E}$ be a collection of i.i.d. symmetric weights, which we call *conductances*. We assume that the conductances are *uniformly elliptic*; that is,

$$\text{there exists a constant } C_M > 0, \text{ such that } \mu_{x,y} \in [C_M^{-1}, C_M] \text{ for all } (x,y) \in E. \tag{1}$$

We say $x \sim y$ if $(x,y) \in E$ and define $\mu_x = \sum_{y\sim x} \mu_{x,y}$. At time 0, consider a Poisson point process of particles on $\mathbb{T}^d$, with intensity measure $\lambda(x) = \lambda_0 \mu_x$ for some constant $\lambda_0 > 0$ and all $x \in \mathbb{T}^d$. That is, for each $x \in \mathbb{T}^d$, the number of particles at x at time 0 is an independent Poisson random variable of mean $\lambda_0 \mu_x$. Then, let the particles perform independent continuous-time simple random walks on the weighted graph so that a particle at $x \in \mathbb{T}^d$ jumps to a neighbor $y \sim x$ at rate $\frac{\mu_{x,y}}{\mu_x}$. It follows from the thinning property of Poisson random variables that the system of particles is in stationarity.

Assume that at time 0 there is an informed particle at the origin, and all other particles are uninformed. One of the main results of this paper is the following.

▶ **Theorem 1.** *If $d \geq 2$ and the conductances satisfy (1), then with probability $1 - n^{-\omega(1)}$ the flooding time is $\Theta(n)$.*

Another main contribution of this paper is the framework we develop to establish Theorem 1, which we believe gives a robust and more easy to apply framework to address problems involving systems of moving particles. The idea is as follows. We tessellate space and time into cells of constant length. Then, for each cell we are given a local event, and call the cell good if the event of that cell holds. Then, if for any given cell, we have that the probability that the cell is good is close enough to 1, then we can find a subset of good cells that form what we call a Lipschitz surface and a Lipschitz net. These Lipschitz surface and Lipschitz net have some percolative and geometric features that allow the good event to propagate through space and time. For example, for the problem of spread of information, the local event we use is to say that a given cell is good if the following two things happen: (i) the cell contains sufficiently many particles, and (ii) if there is an informed particle inside the cell, then that particle is able to inform a large number of other particles that will move

to neighboring cells. With this definition and the existence of the Lipschitz surface and net, we obtain that once the information enters a cell of the Lipschitz surface, we guarantee that the information can propagate throughout the surface, from one cell of the surface to the next. We believe our approach is flexible enough to allow other processes on moving particles to be analyzed. The main task reduces to defining a suitable local event.

Since this framework is quite involved, we will give its construction and all main technical theorems in Section 2. Then, in Section 3, we use this framework to analyze the spread of information. Due to space limitations, we will not be able to give full proofs of the above framework, for which we refer to the full version [6]. This extended abstract has yet one additional result with respect to [6], which is the construction and proof of the Lipschitz net, which is adapted to analyzing processes on finite graphs.

2 Lipschitz net framework

For the remainder of this paper, we assume $d \geq 2$. Fix $\ell > 0$ and tessellate $\mathbb{T}^d$ into cubes of side length $\ell \in \mathbb{R}$, indexed by $i \in \mathbb{Z}^d$. To simplify the notation, assume that n/ℓ is an integer. Next, tessellate time into intervals of length β, indexed by $\tau \in \mathbb{Z}$. With this we denote by the *space-time cell* $(i, \tau) \in \mathbb{Z}^{d+1}$ the region $\prod_{j=1}^{d}[i_j\ell, (i_j + 1)\ell] \times [\tau\beta, (\tau + 1)\beta]$. In the following, β and ℓ are constants such that the ratio β/ℓ^2 is fixed first to be some small value, and then later ℓ is made large enough. We will also need to consider overlapping space-time cells. Let $\eta \geq 1$ be an integer which will represent the amount of overlap between cells. For each cube $i = (i_1, \ldots, i_d)$ and time interval τ, define the *super cube* i as $\prod_{j=1}^{d}[(i_j - \eta)\ell, (i_j + \eta + 1)\ell]$ and the *super interval* τ as $[\tau\beta, (\tau + \eta)\beta]$. We define the *super cell* (i, τ) as the Cartesian product of the super cube i and the super interval τ.

For any time s, let Π_s be the set of particles at time s, seen as a collection of vertices of G with multiplicity when there is more than one particle at a vertex. We say an event E is *increasing* for $(\Pi_s)_{s\geq 0}$ if the fact that E holds for $(\Pi_s)_{s\geq 0}$ implies that it holds for all $(\Pi'_s)_{s\geq 0}$ for which $\Pi'_s \supseteq \Pi_s$ for all $s \geq 0$. We need the following definitions.

▶ **Definition 2.** We say an event E is *restricted* to a region $X \subset \mathbb{T}^d$ and a time interval $[t_0, t_1]$ if it is measurable with respect to the σ-field generated by all the particles that are inside X at time t_0 and their positions from time t_0 to t_1.

▶ **Definition 3.** We say a particle has displacement inside X' during a time interval $[t_0, t_0+t_1]$, if the location of the particle at all times during $[t_0, t_0 + t_1]$ is inside $x + X'$, where x is the location of the particle at time t_0.

▶ **Definition 4.** ν_E is called the *probability associated* to an increasing event E that is restricted to X and a time interval $[0, t]$ if, for an intensity measure ζ and a region $X' \in \mathbb{T}^d$, $\nu_E(\zeta, X, X', t)$ is the probability that E happens given that, at time 0, the particles in X are distributed as a Poisson point process of intensity ζ and their motions from 0 to t are independent continuous time random walks on the weighted graph (G, μ), where the particles are conditioned to have displacement inside X' during $[0, t]$.

For each $(i, \tau) \in \mathbb{T}^d \times \mathbb{Z}$, let $E_{\text{st}}(i, \tau)$ be an increasing event restricted to the super cube i and the super interval τ. Here the subscript st refers to space-time. We say that a cell (i, τ) is *bad* if $E_{\text{st}}(i, \tau)$ does not hold; otherwise, (i, τ) is called *good*.

Our framework will establish that if for any given (i, τ), the event $E_{\text{st}}(i, \tau)$ occurs with large enough probability, then not only do the good cells percolate but the good cells form a particularly useful geometry, which we will call the Lipschitz net.

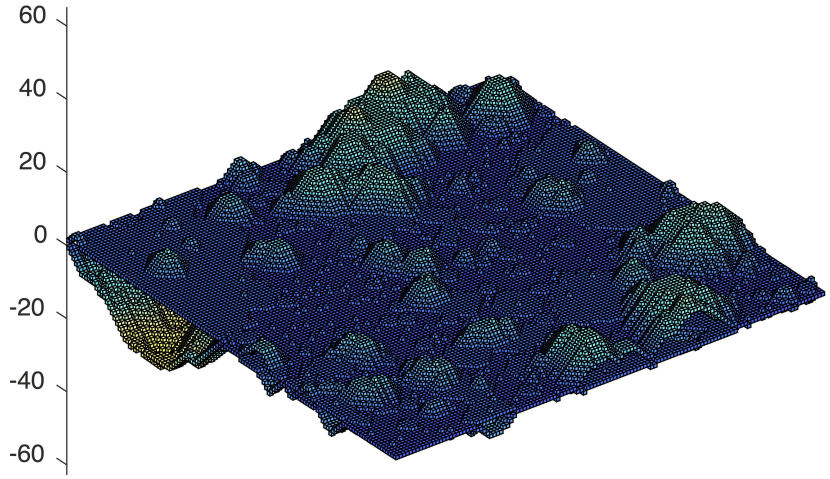


Figure 1 A realization of the two-sided Lipschitz surface for the case $d = 2$.

Before defining the Lipschitz net, we need to introduce a different way to index space-time cells, which we refer to as the *base-height index*. In the base-height index, we pick one of the $d + 1$ space-time dimensions and denote it as *height*, using index $h \in \mathbb{Z}$, while the remaining d space-time dimensions will form the base, which will be indexed by $b \in \mathbb{Z}^d$. In this way, for each space-time cell (i, τ) there will be $(b, h) \in \mathbb{Z}^{d+1}$ such that the base-height cell (b, h) corresponds to the space-time cell (i, τ). With this, we set $E_{\text{bh}}(b, h) = E_{\text{st}}(i, \tau)$. (Here the subscript bh refers to base-height.) It might be tempting to choose time as the height dimension, however it turns out that selecting one of the spatial dimensions to act as height is a better choice, as will be shown below. With this choice, note that $b \in \mathbb{T}^{d-1} \times \mathbb{Z}$ and $h \in \mathbb{T}$; thus, for notation purpose, we define $\mathbb{T}^d_* = \mathbb{T}^{d-1} \times \mathbb{Z}$ and $\mathbb{T}^{d+1}_* = \mathbb{T}^{d-1} \times \mathbb{Z} \times \mathbb{T}$.

2.1 Two-sided Lipschitz surface

▶ **Definition 5.** A function $F : \mathbb{T}^d_* \to \mathbb{T}$ is called a *Lipschitz function* if $|F(x) - F(y)| \leq 1$ whenever $\|x - y\|_1 = 1$.

▶ **Definition 6.** A *two-sided Lipschitz surface* F is a set of base-height cells $(b, h) \in \mathbb{T}^{d+1}_*$ such that for all $b \in \mathbb{T}^d_*$ there are exactly two (possibly equal) integer values $F_+(b) \geq 0$ and $F_-(b) \leq 0$ for which $(b, F_+(b)), (b, F_-(b)) \in F$ and, moreover, F_+ and F_- are Lipschitz functions.

We say a space-time cell (i, τ) belongs to F if its corresponding base-height cell (b, h) belongs to F. For a positive integer D, we say a two-sided Lipschitz surface *surrounds* a cell (b', h') at distance D if any path $(b', h') = (b_0, h_0), (b_1, h_1), \ldots, (b_m, h_m)$ for which $\|(b_i, h_i) - (b_{i-1}, h_{i-1})\|_1 = 1$ for all $i \in \{1, \ldots m\}$ and $\|(b_m, h_m) - (b_0, h_0)\|_1 > D$, intersects with F.

For any $z \in \mathbb{Z}_+$, define the cube $Q_z = [-z/2, z/2]^d$. The following theorem establishes the existence of the Lipschitz surface. Due to space limitations, the proof is given in [6].

▶ **Theorem 7.** *Consider the graph (G, μ) satisfying (1), and the tessellation defined above. There exist positive constants c_1, c_2, c_3, c_4 and c_5 such that, if $\beta/\ell^2 \leq c_5$, then the following holds. Let $E_{\text{st}}(i, \tau)$ be any increasing event restricted to the space-time super cell (i, τ). Fix $\epsilon \in (0, 1)$ and fix $w \geq \sqrt{\frac{\eta\beta}{c_2\ell^2} \log\left(\frac{8c_1}{\epsilon}\right)}$. Then, there exists a positive number α_0 that depends*

on ϵ, η and the ratio β/ℓ^2 so that if

$$\min\left\{C_M^{-1}\epsilon^2\lambda_0\ell^d, \log\left(\frac{1}{1-\nu_{E_{\mathrm{st}}}((1-\epsilon)\lambda, Q_{(2\eta+1)\ell}, Q_{w\ell}, \beta)}\right)\right\} \geq \alpha_0, \tag{2}$$

a two-sided Lipschitz surface F where $E_{\mathrm{st}}(i,\tau)$ holds for all $(i,\tau) \in F$ exists almost surely, and the probability that F does not surround the origin at distance r is at most

$$\begin{cases} \sum_{s\geq r} s^d \exp\left\{-c_3\lambda_0 \frac{\ell s}{\log^{c_4}(\ell s)}\right\}, & \text{for } d = 2 \\ \sum_{s\geq r} s^d \exp\left\{-c_3\lambda_0 \ell s\right\}, & \text{for } d \geq 3. \end{cases}$$

▶ Remark. The proofs in [6] give the existence of the two-sided Lipschitz surface on the whole of $\mathbb{Z}^d$, but the very same proof works for the torus.

▶ Remark. Theorem 7 is key to our framework. We now briefly explain how it can be used. The event E_{st} can be any local event, where in $\nu_{E_{\mathrm{st}}}$, $Q_{(2\eta+1)\ell}$ gives the region on which the event is measurable. To control dependences, we consider the larger cube $Q_{w\ell}$, inside which the particles that start from $Q_{(2\eta+1)\ell}$ are conditioned to stay during the time interval β. Then w has to be large enough, as specified in the theorem, so that this conditioning is likely to happen. Then, $\nu_{E_{\mathrm{st}}}$ gives the probability that the event happens given that the initial configuration of particle is a Poisson point process of intensity measure $(1-\epsilon)\lambda$, just slightly smaller than the intensity measure λ we started with. We disregard an "ϵ-fraction of the particles" because naturally, in any given space-time cell, some particles move atypically and will not be organized exactly as a Poisson point process; but those particles can be neglected using the assumption that E_{st} is increasing. Thus (2) requires that $\nu_{E_{\mathrm{st}}}$ is at least $1-\exp(-\alpha_0)$ for a Poisson point process of intensity $(1-\epsilon)\lambda$. This is usually achievable by properly defining the event E_{st} to be such that its occurrence increases with ℓ (the size of the tessellation). Then, (2) also requires that $C_M^{-1}\epsilon^2\lambda_0\ell^d \geq \alpha_0$. After fixing ϵ, this can be satisfied either by setting ℓ large enough or by assuming that the constant λ_0 governing the density of particles is large enough. This condition is natural in applications: one either requires the size of cells to be large (which will be the case in our application for the flooding time) or the tessellation is more restricted (for example, limited to the transmission radius of the particles) and one requires the density of particles to be large enough, as in [14].

2.2 Lipschitz net

We are now ready to define the *Lipschitz net* on the torus $\mathbb{T}^d$ that we will use to prove Theorem 1. The Lipschitz net, roughly speaking, will be an interlacement of Lipschitz surfaces, where we will take each spatial coordinate as being height in the base-height index, and for each of them we will have a pile of surfaces. More formally, let $k \in \left\{0, 1, \ldots, \left\lfloor \frac{n}{\ell \log^3(n/\ell)} \right\rfloor\right\}$ and $q \in \{1, 2, \ldots, d\}$. For any $i \in \mathbb{T}^d$, let $i = (i_1, i_2, \ldots, i_d)$. Define $\mathbb{L}_k^q$ to be the d-dimensional hyperplane on the space-time tessellation that is orthogonal to the q-th spatial coordinate, with distance from the origin of $k\left\lceil \log^3(n/\ell)\right\rceil$, i.e.

$$\mathbb{L}_k^q = \left\{(i,\tau) \in \mathbb{T}^d \times \mathbb{Z} : \, i_q = k\lceil \log^3(n/\ell)\rceil\right\}.$$

We define F_k^q to be the Lipschitz surface corresponding to $\mathbb{L}_k^q$, i.e. F_k^q is the two-sided Lipschitz surface for which h in the base-height index corresponds to i_q in the space-time index, and for which the Lipschitz functions satisfy $F_+(b) \geq k\lceil \log^3(n/\ell)\rceil$ and $F_-(b) \leq k\lceil \log^3(n/\ell)\rceil$. We define the *height of the surface* F_k^q at $b \in \mathbb{Z}^d$ to be

$$\max_{h:(b,h)\in F_k^q} \left|k\lceil \log^3(n/\ell)\rceil - h\right|.$$

Let $C_0 > 0$ be an integer constant of our choosing. From now on we assume that F_k^q is a Lipschitz surface for which the height is at most $\frac{\log^3(n/\ell)}{2}$ for all $(i,\tau) \in F_k^q$ satisfying $\tau \in \{0,1,\dots,C_0 n/\ell\}$.

▶ **Definition 8.** The *Lipschitz net* F_{net} with constant C_0 is the set of space-time cells $(i,\tau) \in \mathbb{T}^d \times \mathbb{Z}$ contained in the union of all F_k^q; i.e, $F_{\text{net}} = \bigcup_{q=1}^{d} \bigcup_{k=0}^{\left\lfloor \frac{n}{\ell \log^3(n/\ell)} \right\rfloor} F_k^q$. Moreover, we say that F_{net} surrounds the origin at distance D if F_0^q surrounds the origin of $\mathbb{L}_0^q$ at distance D for all $q \in \{1,2,\dots,d\}$.

Note that we have for all $(i,\tau) \in F_{\text{net}}$ that the event $E_{\text{st}}(i,\tau)$ holds, which follows directly from the fact that every space-time cell in F_{net} belongs to at least one Lipschitz surface F_k^q for some k and some q.

▶ **Theorem 9.** *For any constant C_0, there exist a constant $C_1 > 0$ such that, for any $\delta > 0$ and any $\ell = O(n^{1-\delta})$ with $\ell \geq C_1$, the Lipschitz net F_{net} with constant C_0 exists and surrounds the origin at distance $O(\log^2 n)$ with probability $1 - n^{-\omega(1)}$.*

Proof. Start by considering the plane $\mathbb{L}_0^1$ and its corresponding Lipschitz surface F_0^1. If the height of F_0^1 at the origin is more than $\frac{\log^3(n/\ell)}{2}$, then the Lipschitz surface cannot surround the origin at a distance $\frac{\log^3(n/\ell)}{2}$. Therefore, since ℓ and $\log^3(n/\ell)$ are both assumed sufficiently large, we have by Theorem 7 that the probability that a two-sided Lipschitz surface around the origin with height at most $\frac{\log^3(n/\ell)}{2}$ does not exists is at most

$$\sum\nolimits_{s \geq \log^3(n/\ell)/2} s^d \exp\left\{ -C\lambda_0 \frac{\ell s}{\log^c(\ell s)} \right\} \leq \exp\left(-\omega(\log^2 n)\right).$$

Using this and a uniform bound across all space-time cells for which $\tau \in \{0,1,\dots,C_0 n/\ell\}$, we have that the probability that F_0^1 has height at most $\frac{\log^3(n/\ell)}{2}$ for all $(i,\tau) \in F_k^q$ satisfying $\tau \in \{0,1,\dots,C_0 n/\ell\}$, is at least $1 - \exp\left(-\omega(\log^2 n)\right)$.

Next, consider the planes $\mathbb{L}_k^q$. Since the probability space is translation invariant due to the weights $\mu_{x,y}$ being i.i.d., this bound holds for any k and any q. Therefore, by applying a uniform bound across $k \in \left\{0,1,\dots,\left\lfloor \frac{n}{\ell \log^3(n/\ell)} \right\rfloor\right\}$ and $q \in \{1,2,\dots,d\}$ we obtain that the probability that F_k^q has maximum height at most $\frac{\log^3(n/\ell)}{2}$ for all k and q is at least $1 - \exp\left(-\omega(\log^2 n)\right)$. Under this assumption, for any given q and two distinct k, k', the surfaces F_k^q and $F_{k'}^q$ do not intersect, producing the Lipschitz net. ◀

The usefulness of the Lipschitz net is that, once we know it exists for any local event E_{st} that is likely enough, then one just needs to find a suitable choice for the event E_{st} and use the Lipschitz net to show that this event propagates throughout the torus. For the case of spread of information, we will use the Lipschitz net to show that once an informed particle enters a cell that is part of the Lipschitz net, then information spreads evenly across the torus resulting in a density of informed particles. For this, we will use a specific increasing event E_{st} to obtain that the information spreads with positive speed on each individual surface of F_{net}. Then, in order to show that the information also moves across different surfaces of the net, we will need the following geometric property.

▶ **Lemma 10.** *Let F_{net} be the Lipschitz net with constant C_0 and let F_k^q and $F_{k'}^{q'}$ be any two given Lipschitz surfaces that are part of F_{net}, where $q \neq q'$ and $k, k' \in \left\{0,1,\dots,\left\lfloor \frac{n}{\ell \log^3(n/\ell)} \right\rfloor\right\}$. For any $\tau \in \{0,1,\dots,C_0 n/\ell\}$ there exist space-time cells $(i,\tau) \in F_k^q$ and $(i',\tau) \in F_{k'}^{q'}$ such that $\|(i',\tau) - (i,\tau)\|_1 \leq 1$.*

Proof. Let $q = 1$ and $q' = 2$; the proof for other combinations of parameters q and q' goes similarly. We want to show that for any $\tau \in \{0, 1, \ldots, C_0 n/\ell\}$ there exist a space-time cell $(i_1, \ldots, i_d, \tau) \in F^1_k$ and a space-time cell $(j_1, \ldots, j_d, \tau) \in F^2_{k'}$ such that $\|(i_1, \ldots, i_d) - (j_1, \ldots, j_d)\|_1 \leq 1$. Fix $\tau \in \{0, 1, \ldots, C_0 n/\ell\}$ and set the components $(i_3, \ldots, i_d)$ to be the same as $(j_3, \ldots, j_d)$.

Let F^1 be either of the two Lipschitz functions (see Definition 6) corresponding to F^1_k, and let F^2 be either of the Lipschitz functions corresponding to $F^2_{k'}$. Since $(i_3, \ldots, i_d) = (j_3, \ldots, j_d)$, to simplify notation we write $F^1(y) := F^1(y, i_3, \ldots, i_d, \tau) \in \mathbb{T}$ and $F^2(y) := F^2(y, i_3, \ldots, i_d, \tau) \in \mathbb{T}$. Therefore it remains to show that there exists $x, y \in \mathbb{T}$ such that $|(F^1(x), x) - (y, F^2(y))| \leq 1$. Assume, by contradiction, that this is not the case.

Let $m^2 = k' \lceil \log^3(n/\ell) \rceil$, which is the height of $\mathbb{L}^2_{k'}$, that is, $(0, m^2, 0, 0, \ldots, 0) \in \mathbb{L}^2_{k'}$. Next, if F^2 is the Lipschitz function corresponding to F_+ of $F^2_{k'}$ (refer to Definition 6) then set $h^2 = m^2 + \frac{\log^3(n/\ell)}{2} + 1$; otherwise, set $h^2 = m^2 - \frac{\log^3(n/\ell)}{2} - 1$. So for all $y \in \mathbb{T}$, we have $|F^2(y) - m^2| \leq |F^2(y) - h^2|$.

For any point $(x, y) \in \mathbb{T}^2$ we say that it is *under* F^2 if $|y - m^2| \leq |F^2(x) - m^2|$; otherwise we say it is above F^2. Note that $(F^1(m^2), m^2)$ is "under" the surface F^2, and $(F^1(h^2), h^2)$ is above F^2. Therefore, we take the shortest sequence $x_1, x_2, \ldots, x_\iota$ from m^2 to h^2 there must exist a point x_r such that $(F^1(x_r), x_r)$ is under F^2 but $(F^1(x_{r+1}), x_{r+1})$ is above F^2. Since F^1 is Lipschitz, this implies that one of these two points is within distance 1 from F^2. ◀

3 Spread of information using the Lipschitz net

We proceed to showing how the information spreads on F_{net}. We do this by applying Theorem 9 with an event that results in the information spreading with positive speed along each individual Lipschitz surface of F_{net}. More precisely, from now on let the increasing event $E_{\text{st}}(i, \tau)$ be defined as below in Definition 11.

▶ **Definition 11** (Increasing event $E_{\text{st}}(i, \tau)$)**.** Take any $(i, \tau) \in \mathbb{T}^d \times \mathbb{Z}$. Let Υ be the collection of particles located inside $\prod_{j=1}^d [(i_j - \eta)\ell, (i_j + \eta + 1)\ell]$ at time $\tau\beta$. Consider a distinguished particle x_0 located inside $\prod_{j=1}^d [i_j \ell, (i_j + 1)\ell]$ at time $\tau\beta$. Define $E_{\text{st}}(i, \tau)$ to be the event that at time $(\tau + 1)\beta$, for all $i' \in \mathbb{T}^d$ with $\|i - i'\|_\infty \leq \eta$, there is at least one particle from Υ in $\prod_{j=1}^d [i'_j \ell, (i'_j + 1)\ell]$ that collided with x_0 during $[\tau\beta, (\tau + 1)\beta]$.

For $E_{\text{st}}(i, \tau)$ defined as above, we have the following result. The proof of this result uses a few heat-kernel estimates for random walks on $\mathbb{Z}^d$ with i.i.d. conductances, and is given in Section A.

▶ **Lemma 12.** *Fix any ϵ, η and the ratio β/ℓ^2. Let w satisfy the condition in Theorem 7. Then, if ℓ is sufficiently large, then there exists a positive constant C such that for $E_{st}(i, \tau)$ as defined in Definition 11 and for any $(i, \tau) \in \mathbb{T}^d \times \mathbb{Z}$, we have $\nu_{E_{\text{st}}}((1-\epsilon)\lambda, Q_{(2\eta+1)\ell}, Q_{w\ell}, \beta) \geq 1 - \exp\{-C(1 - \epsilon)\lambda_0 \ell^{1/3}\}$.*

Lemma 12 implies that for ℓ large enough, by setting η to be a large enough constant, and defining the increasing event E_{st} as in Definition 11, the information spreads among neighboring cells. Since the Lipschitz net surrounds the origin at distance $O(\log^2 n)$, we have that in at most poly-logarithmic time, the initially informed particle will enter some cell $\prod_{j=1}^d [i_j \ell, (i_j + 1)\ell]$ for which (i, τ) is in some Lipschitz surface F of F_{net}. Once that holds, we know that the event $E_{\text{st}}(i, \tau)$ occurs. By the definition of $E_{\text{st}}(i, \tau)$, we obtain that the initially informed particle in (i, τ) informs other particles causing the information to spread to each $(i', \tau + 1)$ for which $\|i' - i\|_\infty \leq \eta$.

Let (b, h) be the base-height index of the cell $(i, \tau) \in F$. Recall that h is one of the spatial dimensions. We will also select one of the $d-1$ spatial dimensions from b and denote it b_1. Let $b' \in \mathbb{T}_*^d$ be obtained from b by increasing the time dimension from τ to $\tau+1$, and by increasing the chosen spatial dimension from b_1 to b_1+1. Since $\|b-b'\|_1 = 2$, we can choose $h' \in \mathbb{T}$ such that $(b', h') \in F$ and $|h-h'| \leq 2$, where the latter holds by the Lipschitz property of F. Therefore, there must exists $i' \in \mathbb{T}^d$ such that $(i', \tau+1)$ is the space-time cell corresponding to (b', h') and $\|i-i'\|_\infty \leq 4$. Hence, at time $(\tau+1)\beta$, there is an informed particle in the cube indexed by i' if η is at least 4 and $E_{\mathrm{st}}(i, \tau)$ holds.

Using this mechanism, we can show that after some time of order n, the information has spread along the surfaces across the entire torus.

▶ **Lemma 13.** *Let F_{net} be the Lipschitz net with constant C_0 which surrounds the origin at distance $O(\log^2 n)$. There exists a constant $C_T > 0$, independent of C_0, such that for every $(i, \tau) \in F_{net}$ for which $\tau\beta \geq C_T n$, there is at least one informed particle inside the cube $\prod_{j=1}^d [(i_j - \eta + 1)\ell, (i_j + \eta)\ell]$ for all times in $[\tau\beta, (\tau+1)\beta]$.*

Proof. Let $E_{\mathrm{st}}(i, \tau)$ be defined as in Lemma 12 and let F_{net} be the Lipschitz net with constant C_0, corresponding to the event $E_{\mathrm{st}}(i, \tau)$. We have by the fact that F_{net} surrounds the origin at a distance $O(\log^2 n)$ and that each cell represents a time interval of length β, that it takes at most $O(\beta \log^2 n)$ time for the information to enter F_{net}. Once the informed particle is in a space-time cell of some surface F_k^q of F_{net}, we have by the definition of $E_{\mathrm{st}}(i, \tau)$ with $\eta = d$, that it takes at most $\frac{2n}{\ell}$ steps for the information to spread across the surface (moving between neighboring cells), so that all space-time cells $(i, \tau) \in F_k^q$ for which $\tau = \frac{2n}{\ell} + O(\log^2 n)$ contain an informed particle.

Next, for any q', k' with $q' \neq q$, we know by Lemma 10 that for any τ there are neighboring cells $(i, \tau) \in F_k^q$ and $(i', \tau) \in F_{k'}^{q'}$. Therefore, it takes at most β time for the information to enter any surface $F_{k'}^{q'}$ with $q' \neq q$, and another $\frac{2n}{\ell}\beta$ amount of time to spread to all cells in those surfaces, so that all cells $(i, \tau) \in F_{k'}^{q'}$ for which $\tau = \frac{4n}{\ell} + 1 + O(\log^2 n)$ contains an informed particle. It still remains to spread the information to the surfaces $F_{k'}^q$ with $k' \neq k$. Again, this takes at most $\frac{2n}{\ell}\beta + \beta$ time by the same argument above. Putting everything toghether, we obtain that for any k, any q, and all $(i, \tau) \in F_k^q$ for which $\tau\beta \geq C_T n \geq (\frac{6n}{\ell} + 2 + O(\log^2 n))\beta$, where we set C_T large enough for the second inequality to hold, there is at least one informed particle in the cube $\prod_{j=1}^d [(i_j - \eta + 1)\ell, (i_j + \eta)\ell]$ for all times in $[\tau\beta, (\tau+1)\beta]$. ◀

Using Lemma 13 and the geometric properties of the Lipschitz net, we can show that there is a density of informed particles everywhere on the torus for an interval of time of order n.

▶ **Theorem 14.** *There exists constants $C_\beta \geq 1$ and $C_\ell > 0$ such that the following holds. Let C_T be the constant from Lemma 13. Tessellate $\mathbb{T}^d$ into cubes $(Q_m)_m$ of side length $C_\ell \log^3(n)$. Then, for all times $t \in [C_T n, (C_T + C_\beta)n]$, there is at least one informed particle in each subcube Q_m with probability at least $1 - n^{-\omega(1)}$.*

Proof. Fix ℓ sufficiently large for Lemma 12 and Theorem 7 to hold and recall that the ratio β/ℓ^2 is fixed. Let also $n \gg \ell$. Then, there exists a constant C_T so that, for any large enough choice of C_0, Lemma 13 gives that for every space-time cell (i, τ) of the Lipschitz net F_{net} that satisfies $\tau\beta \geq C_T n$, there is at least one informed particle in the region $\prod_{j=1}^d [(i_j - \eta + 1)\ell, (i_j + \eta)\ell]$ at all times in $[\tau\beta, (\tau+1)\beta]$. We can, without loss of generality, assume C_T is such that $C_T n = \beta\tau^*$ for some $\tau^* \in \mathbb{N}$. Then, we only

have to show that for all cubes Q_m of side length $C_\ell \log^3(n)$, there exist space-time cells (i,τ) such that the region $\prod_{j=1}^d [(i_j-\eta+1)\ell,(i_j+\eta)\ell]$ is contained in Q_m and such that $[C_T n,(C_T+C_\beta)n] \subseteq \bigcup_\tau [\tau\beta,(\tau+1)\beta]$, where C_β is a constant greater or equal to 1.

Let $C_\beta = k^*\beta$ where k^* is the smallest integer for which $k^*\beta \geq 1$ and fix the Lipschitz net constant C_0 to be greater or equal to $(C_T+C_\beta)\ell/\beta$. Then, we have from Theorem 9 that the Lipschitz net with constant C_0 exists with probability at least $1-n^{-\omega(1)}$.

We now show that if this Lipschitz net exists, the lemma holds. Let F_s^q and F_{s+1}^q be any two consecutive two-sided surfaces of the Lipschitz net and let $(b,h) \in F_s^q$ and $(b,h') \in F_{s+1}^q$ be two base-height cells with the same base. By definition of the Lipschitz net, we have that the height of each Lipschitz surface in the net is at most $\frac{\log^3(n/\ell)}{2}$ for all space-time cells that satisfy $\tau \in \{0,1,\dots,C_0 n/\ell\}$. Since the base-height cells (b,h) and (b,h') might belong to opposite sides of the two-sided Lipschitz surfaces, we therefore have that $|h-h'| \leq 2\log^3(n/\ell)$ for all base-height cells for which $\tau\beta < (C_T+C_\beta)n \leq C_0\beta n/\ell$. Note that this holds for all $q \in \{1,\dots,d\}$ and recall that by Lemma 13 there is an informed particle inside the region $\prod_{j=1}^d [(i_j-\eta+1)\ell,(i_j+\eta-1)\ell]$ throughout the entire time interval $[\tau\beta,(\tau+1)\beta]$. Therefore, for every cube of side length at least $2\ell\log^3(n/\ell)+2\eta\ell$ on the torus and throughout every time interval of the form above, there is at least one informed particle inside the cube. By repeating this argument for all τ that satisfy $\tau\beta \in [C_T n,(C_T+C_\beta)n)$, we have that this holds for the entire time interval $[C_T n,(C_T+C_\beta)n]$. ◀

Before turning to the proof of Theorem 1, we state a theorem that gives that if we start with a density of particles on a cube, regardless of how they are placed inside some subcubes, we can couple their positions after some time with a Poisson point process that is independent of their initial locations. This gives a type of local mixing property for random walks on $\mathbb{T}^d$ with i.i.d. conductances. For the proof of this technical result, refer to [7, Theorem 3.1].

▶ **Theorem 15.** *Let G be a uniformly elliptic graph with edge weights $\mu_{x,y}$. There exist constants c_0, c_1, $C>0$ such that the following holds. Fix $K>\ell>0$ and $\epsilon \in (0,1)$. Consider the cube Q_K tessellated into subcubes $(T_i)_i$ of side length ℓ and assume that ℓ is large enough. Let $(x_j)_j \subset Q_K$ be the locations at time 0 of a collection of particles, such that each subcube T_i contains at least $\sum_{y\in T_i}\beta\mu_y$ particles for some $\beta>0$. Let $\Delta \geq c_0\ell^2\epsilon^{-4/\Theta}$ where Θ is a constant that depends on the weight bounds. For each j denote by Y_j the location of the j-th particle at time Δ. Fix $K'>0$ such that $K-K' \geq \sqrt{\Delta}c_1\epsilon^{-1/d}$. Then there exists a coupling $\mathbb{Q}$ of an independent Poisson point process ψ with intensity measure $\zeta(y)=\beta(1-\epsilon)\mu_y$, $y \in \mathcal{C}_\infty$, and $(Y_j)_j$ such that within $Q_{K'} \subset Q_K$, ψ is a subset of $(Y_j)_j$ with probability at least*

$$1-\sum_{y\in Q_{K'}} \exp\left\{-C\beta\mu_y\epsilon^2\Delta^{d/2}\right\}.$$

3.1 Proof of Theorem 1

Proof. Let C_T be the constant from Lemma 13 and let C_β and C_ℓ be the constants from Theorem 14. We want to bound the probability that at time $(C_T+C_\beta)n$ there is at least one particle on $\mathbb{T}^d$ that is not informed. By using that the particles on $\mathbb{T}^d$ form a Poisson point process with intensity $\lambda(y)=\lambda_0\mu_y$, we have that this probability can be bounded from above by $\lambda_0 p_n \sum_{y\in\mathbb{T}^d}\mu_y$,where p_n is an upper bound for the probability that a single particle is not informed by time $(C_T+C_\beta)n$ on the torus of side length n, uniformly on the initial location of the particle. We now proceed to find the bound p_n.

Let F be the event that a particle located somewhere on the torus does not become informed during $[C_T n,(C_T+C_\beta)n]$. Note that the probability that a particle does not

get informed by time $(C_T + C_\beta)n$ is smaller than the probability of F, so $p_n \leq \mathbb{P}[F]$. Let $t \in (0, C_\beta n)$ be a time step we will fix later and consider the time interval $[C_T n, (C_T + C_\beta)n]$ split into subintervals of length t, i.e. let the interval be split into subintervals of the form $[C_T n + kt, C_T n + (k+1)t]$ for $k \in \{0, 1, \ldots, \lfloor C_\beta n/t \rfloor - 1\}$. Let F_k denote the event that a particle located somewhere on the torus does not become informed during the time interval $[C_T n + kt, C_T n + (k+1)t]$. We then have that

$$\mathbb{P}[F] \leq \mathbb{P}[F_0 \cap F_1 \cap \cdots \cap F_{\lfloor C_\beta n/t \rfloor - 1}].$$

Tessellate $\mathbb{T}^d$ into cubes $(Q_i)_i$ of side length $C_\ell \log^3(n)$, indexed by i. Let D_k be the event that time $C_T n + kt$ there is at least one informed particle in every cube Q_i. We can then write

$$\mathbb{P}[F_0 \cap F_1 \cap \cdots \cap F_{\lfloor C_\beta n/t \rfloor - 1}] \leq \mathbb{P}\left[\bigcap_{k=0}^{\lfloor C_\beta n/t \rfloor - 1} (F_k \cap D_k)\right] + \mathbb{P}\left[\bigcup_{k=0}^{\lfloor C_\beta n/t \rfloor - 1} D_k^{\mathsf{c}}\right]. \tag{3}$$

To bound the second term, we apply Theorem 14, which gives that there is at least one informed particle in every cube Q_i of side length $C_\ell \log^3(n)$ for all times during $t \in [C_T n, (C_T + C_\beta)n]$ with high probability. Therefore, it holds that

$$\mathbb{P}\left[\bigcup_{k=0}^{\lfloor C_\beta n/t \rfloor - 1} D_k^{\mathsf{c}}\right] = n^{-\omega(1)}. \tag{4}$$

We now focus on the first term of (3). By rearranging the expression inside the probability and using the chain rule, we have that

$$\mathbb{P}\left[\left(\bigcap_{k=0}^{\lfloor C_\beta n/t \rfloor - 1} F_k\right) \cap \left(\bigcap_{k=0}^{\lfloor C_\beta n/t \rfloor - 1} D_k\right)\right] \leq \mathbb{P}[F_0 \cap D_0] \prod_{k=1}^{\lfloor C_\beta n/t \rfloor - 1} \mathbb{P}\left[F_k \cap D_k \,\middle|\, \bigcap_{j<k} F_j \cap D_j\right].$$

In order to bound the terms $\mathbb{P}\left[F_k \cap D_k \,\middle|\, \bigcap_{j<k} F_j \cap D_j\right]$, first note that

$$\begin{aligned}\mathbb{P}\left[F_k \cap D_k \,\middle|\, \bigcap_{j<k} F_j \cap D_j\right] &\leq \mathbb{P}\left[F_k \,\middle|\, D_k \cap \bigcap_{j<k} F_j \cap D_j\right] \mathbb{P}\left[D_k \,\middle|\, \bigcap_{j<k} F_j \cap D_j\right] \\ &\leq \mathbb{P}\left[F_k \,\middle|\, D_k \cap \bigcap_{j<k} F_j \cap D_j\right],\end{aligned}$$

and similarly, $\mathbb{P}[F_0 \cap D_0] = \mathbb{P}[F_0 \mid D_0]\mathbb{P}[D_0] \leq \mathbb{P}[F_0 \mid D_0]$.

Next, we show a bound for $\mathbb{P}\left[F_k \,\middle|\, D_k \cap \bigcap_{j<k} F_j \cap D_j\right]$ that holds uniformly on all configurations for which D_k holds. We do this by applying Theorem 15 to find a uniform bound on the probability of a particle remaining uninformed, given there is a density of informed particles on the torus $\mathbb{T}^d$ at the beginning of the time interval we consider. More precisely, we set the terms of Theorem 15 as follows, where we mark them with a bar to help distinguish them from other terms in this proof. Let $\bar{K} = n$, $\bar{\ell} = C_\ell \log^3(n)$, and $\bar{\epsilon} = \frac{1}{2}$. Let $\bar{\Delta} = C_\Theta \log^8(n)$, where C_Θ is a constant sufficiently large for $\bar{\Delta}$ to satisfy the conditions of Theorem 15 for all n. We fix the time step t to be equal to $\bar{\Delta}$ and let $\bar{K}' = n - C_{\bar{\epsilon}}\sqrt{\bar{\Delta}}$, where $C_{\bar{\epsilon}} = c_1 \bar{\epsilon}^{-1/d}$. We now have by the definition of D_k for every $k \in \{0, 1, \ldots, \lfloor C_\beta n/\bar{\Delta} \rfloor - 1\}$

that at time $C_T n + kt$ there is at least one informed particle in every subcube Q_i, so there are at least

$$\frac{1}{C_M d C_\ell^d \log^{3d}(n)} \sum_{y \in Q_i} \mu_y$$

informed particles in every cube. We set the parameter $\bar{\beta}$ from Theorem 15 to be $\bar{\beta} = \frac{1}{C_M d C_\ell^d \log^{3d}(n)}$ and apply the theorem. This gives us that after the informed particles move around for time $\bar{\Delta}$, they stochastically dominate a Poisson point process of intensity $\bar{\zeta}(y) = \frac{1}{2}\frac{1}{C_M d C_\ell^d \log^{3d}(n)}\mu_y$ inside the cube of side length $\bar{K}'$. Using (1), we have that this coupling fails with probability at most

$$\sum_{y \in Q_{K'}} \exp\left\{-C\frac{1}{4}\frac{1}{C_M d C_\ell^d \log^{3d}(n)} C_\Theta^{d/2} \log^{4d}(n)\mu_y\right\} \leq n^d \exp\{-C_1 \log^d(n)\}, \tag{5}$$

where C is the constant from Theorem 15 and C_1 is some constant that depends on d. Note that this bound only depends on the size of $Q_{K'}$ and as such is independent of the site the cube is centered around.

Next, if D_k holds and the coupling succeeds, the number of informed particles at a given site y of the torus at time $C_T n + (k+1)t$ stochastically dominates a Poisson random variable of intensity $\frac{1}{2C_M d C_\ell^d \log^{3d}(n)}\mu_y$. Since the probability that a particle is not informed during the interval $[C_T n + kt, C_T n + (k+1)t]$ is smaller than the probability of not getting the information only at the end of the interval, we have that $\mathbb{P}[F_k \mid \{\text{coupling succeeds}\} \cap D_k]$ can be bound by the probability that at the end of the time interval, there are no informed particles at the location of the particle we are considering. Using (1) to bound μ_y, we have for some constant C_2 that $\mathbb{P}[F_k \mid \{\text{coupling succeeds}\} \cap D_k]$ is at most the probability that a Poisson random variable with intensity $\frac{C_2}{\log^{3d}(n)}$ is 0, i.e.

$$\mathbb{P}[F_k \mid \{\text{coupling succeeds}\} \cap D_k] \leq \exp\left\{-\frac{C_2}{\log^{3d}(n)}\right\}. \tag{6}$$

This bound holds uniformly across all sites of the torus where the particle might be located and across all configurations of particles for which D_k holds. Combining (5) and (6) we therefore have for all $k \in \{0, 1, \ldots, \lfloor C_\beta n/t \rfloor - 1\}$ that

$$\mathbb{P}\left[F_k \;\middle|\; D_k \cap \bigcap_{j<k} F_j \cap D_j\right] \leq n^d \exp\{-C_1 \log^d(n)\} + \exp\left\{-\tfrac{C_2}{\log^{3d}(n)}\right\}.$$

Using the definition of t, the bound from (4) and applying the above bound for all $k \in \{0, 1, \ldots, \lfloor C_\beta n/t \rfloor - 1\}$, we have that $\mathbb{P}[F_0 \cap F_1 \cap \cdots \cap F_{\lfloor C_\beta n/t \rfloor - 1}]$ from (3) is smaller than

$$\left(n^d \exp\{-C_1 \log^d(n)\}\right)^{C_\beta n/(C_\Theta \log^8(n))} + \exp\left\{-\tfrac{C_2 C_\beta n}{C_\Theta \log^{3d+8}(n)}\right\} + n^{-\omega(1)}.$$

Using that $p_n \leq \mathbb{P}[F_0 \cap F_1 \cap \cdots \cap F_{\lfloor C_\beta n/t \rfloor - 1}]$ and $\mu_y \leq C_M d$ by (1), we get that the probability that there exists a particle that has not been informed by time $(C_T + C_\beta)n$ is at most

$$C_M d \lambda_0 n^d \left(\left(n^d \exp\{-C_1 \log^d(n)\}\right)^{C_\beta n/(C_\Theta \log^8(n))} + \exp\left\{-\tfrac{C_2 C_\beta n}{C_\Theta \log^{3d+8}(n)}\right\} + n^{-\omega(1)}\right).$$

Since the above is $n^{-\omega(1)}$, the proof is completed. ◀

4 Conclusion

We have established a tight bound on the flooding time (up to constant factors) for the spread of information between random walk particles on the discrete torus of size n, equipped with i.i.d., uniformly elliptic conductances. To prove this, we develop a framework to control dependences, which given any increasing, local event that is likely enough, one can find a Lipschitz surface and a Lipschitz net through space-time where this event holds. We believe this result can be applicable to analyze other processes and algorithms on systems of random walk particles. We also believe that this framework can be adapted to work with different types of particle systems, for example, when the particles do not move independently of one another, but nonetheless obey some local mixing.

References

1 Martin T. Barlow. Random walks on supercritical percolation clusters. *Annals of Probability*, 32(4):3024–3084, 2004. doi:10.1214/009117904000000748.

2 Martin T. Barlow and Ben M. Hambly. Parabolic Harnack inequality and local limit theorem for percolation clusters. *Electronic Journal of Probability*, 14:1–26, 2009. doi:10.1214/EJP.v14-587.

3 Elisabetta Candellero and Augusto Teixeira. Percolation and isoperimetry on transitive graphs. *arXiv*, 2015. arXiv:1507.07765.

4 Andrea Clementi, Angelo Monti, Francesco Pasquale, and Riccardo Silvestri. Information Spreading in Stationary Markovian Evolving Graphs. *IEEE Transactions on Parallel and Distributed Systems*, 22(9):1425–1432, 2011. doi:10.1109/TPDS.2011.33.

5 Andrea E. F. Clementi, Francesco Pasquale, and Riccardo Silvestri. MANETS: High mobility can make up for low transmission power. *IEEE/ACM Transactions on Networking*, 21(2):610–620, 2009. doi:10.1109/TNET.2012.2204407.

6 Peter Gracar and Alexandre Stauffer. Multi-scale Lipschitz percolation of increasing events for Poisson random walks. *arXiv*, 2017. arXiv:1702.08748.

7 Peter Gracar and Alexandre Stauffer. Random walks in random conductances: decoupling and spread of infection. *arXiv*, 2017. arXiv:1701.08021.

8 Harry Kesten and Vladas Sidoravicius. The spread of a rumor or infection in a moving population. *Annals of Probability*, 33(6):2402–2462, 2005. doi:10.1214/009117905000000413.

9 Henry Lam, Zhenming Liu, Michael Mitzenmacher, Xiaorui Sun, and Yajun Wang. Information Dissemination via Random Walks in d-Dimensional Space. *arXiv*, 2011. arXiv:1104.5268.

10 T. M. Liggett, R. H. Schonmann, and A. M. Stacey. Domination by product measures. *Ann. Probab.*, 25(1):71–95, 01 1997. doi:10.1214/aop/1024404279.

11 Yuval Peres, Alistair Sinclair, Perla Sousi, and Alexandre Stauffer. Mobile geometric graphs: Detection, coverage and percolation. *Probability Theory and Related Fields*, 156(1-2):273–305, 2013. doi:10.1007/s00440-012-0428-1.

12 Alberto Pettarin, Andrea Pietracaprina, Geppino Pucci, and Eli Upfal. Tight bounds on information dissemination in sparse mobile networks. In *Proceedings of the 30th Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing*, PODC '11, pages 355–362, New York, NY, USA, 2011. ACM. doi:10.1145/1993806.1993882.

13 Vladas Sidoravicius and Alain-Sol Sznitman. Percolation for the vacant set of random interlacements. *Communications on Pure and Applied Mathematics*, 62(6):831–858, 2009. doi:10.1002/cpa.20267.

14 Alexandre Stauffer. Space-time percolation and detection by mobile nodes. *Annals of Applied Probability*, 25(5):2416–2461, 2015. doi:10.1214/14-AAP1052.

15 Alain-Sol Sznitman. Decoupling inequalities and interlacement percolation on $G \times \mathbb{Z}$. *Inventiones mathematicae*, 187(3):645–706, 2012. doi:10.1007/s00222-011-0340-9.

A Appendix: Proof of Lemma 2

Let $T = \ell^{5/3}$. Since β/ℓ^2 is fixed, we can set ℓ a large enough constant so that $T \ll \beta$ (i.e., T is much smaller than the length of the time interval in the tessellation). Define $Q^* := \prod_{j=1}^d [(i_j - \eta)\ell, (i_j + \eta + 1)\ell]$ and assume that at time $\tau\beta$, for all sites $x \in Q^*$, the number of particles at x is a Poisson random variable with mean $(1-\epsilon)\lambda_0\mu_x$.

We start by stating two two claims and using them to prove the lemma. Then, we give the proof of the claims.

▶ **Claim 16.** *If the distinguished particle x_0 is inside $\prod_{j=1}^d [i_j\ell, (i_j+1)\ell]$ at time $\tau\beta$ and x_0 follows a fixed path $(\rho(t))_{\tau\beta \le t \le \tau\beta+T}$, then by time $\tau\beta + T$ the number of particles that have collided with x_0 during $[\tau\beta, \tau\beta + T]$, but were not at the same site as x_0 at time $\tau\beta$, is a Poisson random variable with intensity at least $C_1(1-\epsilon)\lambda_0\ell^{1/3}$ for some positive constant C_1, independent of $\rho(t)$.*

▶ **Claim 17.** *Given that there are N particles inside of Q^* at time $\tau\beta + T$, the probability that at least one of these particles is inside $Q^{**} := \prod_{j=1}^d [(i'_j)\ell, (i'_j+1)\ell]$ for any i' for which $|i - i'| \le \eta$ is at least $1 - \exp\{-Nc_p\}$, where c_p is a positive constant that is bounded away from 0 and depends only on d, η and the ratio β/ℓ^2.*

Now we use the above claims to prove the lemma. Note that by Definition 11, $E_{\text{st}}(i,\tau)$ is restricted to the super cube Q^* and time interval $[\tau\beta, (\tau+1)\beta]$. We now define the following 3 events.

F_1: The distinguished particle x_0 never leaves $\prod_{j=1}^d [(i_j - \eta + 1)\ell, (i_j + \eta - 1)\ell]$ during $[\tau\beta, \tau\beta + T]$.

F_2: Let C_1 be the constant from Claim 16. During the time interval $[\tau\beta, \tau\beta + T]$ the distinguished particle x_0 collides with at least $\frac{C_1\lambda_0\ell^{1/3}}{2}$ different particles from Υ that are in the super cube Q^* at time $\tau\beta + T$.

F_3: Out of the $\frac{C_1\lambda_0\ell^{1/3}}{2}$ or more particles from F_2, at least one of them is in the cube Q^{**} at time $(\tau+1)\beta$, for all Q^{**} for which $Q^{**} \subset Q^*$.

By definition of the events, we clearly have that $\mathbb{P}[E_{\text{st}}(i,\tau)] \ge \mathbb{P}[F_1 \cap F_2 \cap F_3]$. Also note that F_1, F_2 and F_3 are clearly restricted to the super cube Q^* and the time interval $[\tau\beta, (\tau+1)\beta]$ and are all increasing events.

Using the exit probability bound from [1, Proposition 3.7] we have

$$\mathbb{P}[F_1] \ge 1 - C_2 \exp\{-C_3\ell^2/T\} = 1 - C_2\exp\{-C_3\ell^{1/3}\} \tag{7}$$

for some positive constants C_2 and C_3.

For the event F_2, we apply the result of Claim 16. Note that the bound from Claim 16 is uniform across all paths $\rho(\cdot)$ and in particular holds for any path the distinguished particle from the event F_1 might follow. This gives that the intensity of the Poisson point process of particles that are in Q^* at time $\tau\beta$ and collide with x_0 during $[\tau\beta, \tau\beta + T]$ is at least $(1-\epsilon)\lambda_0 C_1\ell^{1/3}$ for some positive constant C_1. Since every particle that collides with x_0 enters $\prod_{j=1}^d [(i_j - \eta + 1)\ell, (i_j + \eta)\ell]$ during $[\tau\beta, \tau\beta + T]$, we can again use the exit probability bound from [1, Proposition 3.7] to bound the probability that the particle is outside of Q^* at time $\tau\beta + T$ from below by

$$1 - C_a \exp\left\{-\frac{C_b\ell^2}{T}\right\} = 1 - C_a\exp\{-C_b\ell^{1/3}\},$$

for some positive constants C_a and C_b. This term can be made as close to 1 as possible by having ℓ sufficiently large. We assume ℓ is large enough so that this term is larger than $2/3$.

This gives that the intensity of the process of particles from Υ that collided with x_0 during $[\tau\beta, \tau\beta + T]$ and are in Q^* at time $\tau\beta + T$ is at least

$$\frac{2(1-\epsilon)\lambda_0 C_1 \ell^{1/3}}{3}.$$

Using Chernoff's bound (see Lemma 18) we have that

$$\mathbb{P}[F_2] \geq 1 - \exp\{-(2/3)^2 C_1(1-\epsilon)\lambda_0 \ell^{1/3}\}. \tag{8}$$

We now turn to F_3. Using the result of Claim 17, and a uniform bound across the number of cubes inside a super cube, we have that

$$\mathbb{P}[F_3] \geq 1 - (2\eta+1)^d \exp\left\{-\frac{C_1(1-\epsilon)\lambda_0 \ell^{1/3}}{2} c_p\right\}, \tag{9}$$

where c_p is a small but positive constant. Taking the product of the probability bounds in (7), (8) and (9), we see that the probability that $E_{\mathrm{st}}(i,\tau)$ holds is at least

$$1 - \exp\{-C(1-\epsilon)\lambda_0 \ell^{1/3}\}$$

for some constant C and all large enough ℓ, which proves the claim.

Proof of Claim 16. For each time $t \in [\tau\beta, \tau\beta + T]$, let Ψ_t be the Poisson point process on $\mathbb{T}^d$ giving the locations at time t of the particles that belong to Υ, excluding all particles located at $\rho(\tau\beta)$ at time $\tau\beta$. Since the particles that start in Q^* move around and can leave Q^*, we need to find a lower bound for the intensity of Ψ_t for times in $[\tau\beta, \tau\beta + T]$. Note that the distinguished particle x_0 we are tracking is not part of Ψ, since Ψ does not include particles located at $\rho(\tau\beta)$ at time $\tau\beta$.

We will need to apply heat kernel bounds from [2, Theorem 2.2] to the particles in Q^*, so we need to ensure that the time intervals we consider are large enough for the bounds to hold. We will only consider times $t \in [\ell^{4/3}, T]$ so that for large enough ℓ, we have $t \geq \sup_{\substack{x \in Q^* \\ y \in Q^*}} \|x - y\|_1$ and so the heat kernel bounds from [2, Theorem 2.2] hold. Then, we have that for all sites $x \in Q^*$ that are at least ℓ away from the boundary of Q^* and at any such time t the intensity of $\Psi_{\tau\beta+t}$ at vertex $x \in \mathbb{T}^d$ is at least

$$\Psi_{\tau\beta+t}(x) \geq \sum_{\substack{y \in Q^* \\ y \neq \rho(\tau\beta)}} (1-\epsilon)\lambda_0 \mu_y \cdot \mathbb{P}_y[Y_t = x] = (1-\epsilon)\lambda_0 \mu_x \sum_{\substack{y \in Q^* \\ y \neq \rho(\tau\beta)}} \mathbb{P}_x[Y_t = y],$$

where Y_t stands for the location of a simple random walk at time t, and $\mathbb{P}_y$ is the measure induced by a simple random walk starting from y. In the last step above, we used that the simple random walk is reversible with respect to the measure μ. We now use the exit probability bound from [1, Proposition 3.7] to get that

$$\sum_{y \in Q^*} \mathbb{P}_x[Y_t = y] \geq 1 - c_3 \exp\{-c_4 \ell^2/t\}.$$

Next, we use [2, Theorem 2.2] to account for the particles at $\rho(\tau\beta)$, yielding

$$\sum_{\substack{y \in Q^* \\ y \neq \rho(\tau\beta)}} \mathbb{P}_x[Y_t = y] \geq 1 - c_3 \exp\left\{-c_4 \ell^2/t\right\} - C_M c_5 t^{-d/2}.$$

This gives that for any $t \in [\ell^{4/3}, T]$, the intensity of $\Psi_{\tau\beta+t}$ is at least

$$\Psi_{\tau\beta+t}(x) \geq (1-\epsilon)\lambda_0\mu_x(1 - c_3\exp\{-c_4\ell^2/T\} - C_M c_5 \ell^{-2d/3}).$$

Let $[\tau\beta, \tau\beta+T]$ be divided into subintervals of length $W \in (0, T]$, where we set $W = \ell^{4/3}$ so that it is large enough to allow the use of the heat kernel bounds from [2, Theorem 2.2]. Let $J = \{1, \ldots, \lfloor T/W \rfloor\}$ and $t_j := \tau\beta + jW$. Then the intensity of particles that share a site with the distinguished particle x_0 at least once among times $\{t_1, t_2, \ldots, t_{\lfloor T/W \rfloor}\}$ is at least

$$\begin{aligned}
&\sum_{j\in J} \Psi_{t_j}(\rho(t_j))\mathbb{P}_{\rho(t_j)}[Y_{r-t_j} \neq \rho(r)\ \forall r \in \{t_{j+1}, \ldots, t_{\lfloor T/W \rfloor}\}] \\
&\geq (1-\epsilon)\lambda_0 C_M^{-1}(1 - c_3\exp\{-c_4\ell^2/T\} - C_M c_5\ell^{-2d/3}) \sum_{j\in J}\left(1 - \sum_{z>j}\mathbb{P}_{\rho(t_j)}[Y_{t_z-t_j} = \rho(t_z)]\right).
\end{aligned}$$

We want to make all of the terms of the sum over J positive, so we consider the term $\sum_{z>j}\mathbb{P}_{\rho(t_j)}[X_{t_z-t_j} = \rho(t_z)]$ and show that it is smaller than $\frac{1}{2}$ for large enough ℓ. To do this, we use [2, Theorem 2.2], which hold when $W \geq \ell^{4/3}$ and ℓ is large enough, to bound it from above by

$$\begin{aligned}
\sum_{z>j}\mathbb{P}_{\rho(t_j)}[Y_{t_z-t_j} = \rho(t_z)] &\leq \sum_{z>j} C_M C_{HK}(t_z - t_j)^{-d/2} \\
&\leq C_M C_{HK} W^{-d/2} \sum_{z=1}^{T/W-j} z^{-d/2} \qquad (10)
\end{aligned}$$

where C_{HK} is a constant coming from [2, Theorem 2.2]. Then, (10) can be bounded from above by

$$C_M C_{HK} W^{-d/2}\left(2 + \sum_{z=3}^{T/W-j} z^{-d/2}\right) \leq C_M C_{HK} W^{-d/2}\left(2 + \int_2^{T/W} z^{-d/2}dz\right). \qquad (11)$$

Let C be a constant that can depend on C_{HK}, C_M and d. Then for $d = 2$, (11) is smaller than $CW^{-1}\log(T/W)$, and for $d \geq 3$ the expression in (11) is smaller than $CW^{-d/2}$. Thus, setting ℓ large enough, both terms are smaller than $\frac{1}{2}$.

Then, as a sum of Poisson random variables, we get that Υ' is a Poisson random variable with a mean at least

$$(1-\epsilon)\lambda_0 C_M^{-1}(1 - c_3\exp\{-2c_4\ell^2/T\} - C_M c_5\ell^{-2d/3})\tfrac{T}{2W}.$$

Using that $T = \ell^{5/3}$ and setting ℓ large enough establishes the claim, with C_1 being any constant satisfying $C_1 < \frac{C_M^{-1}}{2}$. ◀

Proof of Claim 17. We now prove that for large enough ℓ, if there are N particles inside of Q^* at time $\tau\beta + T$, there is at least one of them inside Q^{**} at time $(\tau+1)\beta$ with probability at least $1 - \exp\{-Nc_p\}$.

For $t^{2/3} \geq \sup_{\substack{x\in Q^* \\ y\in Q^{**}}} \|x - y\|_1$, define $p_t := \inf_{x\in Q^*}\sum_{y\in Q^{**}}\mathbb{P}_x[Y_t = y]$. Then, if we define $\mathrm{bin}(N, p_t)$ to be a binomial random variable with parameters $N \in \mathbb{N}$ and $p_t \in [0, 1]$, it directly follows that we can bound probability of one of the N particles from Q^* being inside Q^{**} at time $(\tau+1)\beta$ from below by

$$\mathbb{P}[\mathrm{bin}(N, p_t) \geq 1] \geq 1 - \exp\{-Np_t\}.$$

It remains to show that for $t = \beta - T$, we have that $p_t \geq c_p > 0$ for some constant c_p. We will again use the heat kernel bounds from [2, Theorem 2.2] for the pair x, y, which hold if $\|x-y\|_1^{3/2} \leq \beta - T$ for all $x \in Q^*, y \in Q^{**}$. Given the ratio β/ℓ^2, d and η, this is satisfied if ℓ is large enough. Then we have that

$$p_{\beta-T} = \inf_{x\in Q^*} \sum_{y\in Q^{**}} \mathbb{P}_x[Y_{\beta-T} = y] \geq \inf_{x\in Q^*} C_M^{-1} \sum_{y\in Q^{**}} c_1 \beta^{-d/2} \exp\left\{-c_2 \frac{\|x-y\|_1^2}{\beta - T}\right\}.$$

Now we use that x and y can be at most $c_\eta \ell$ apart where c_η is a constant depending on d and η only, and that $\beta - T \geq \beta/2$ for ℓ large enough. Hence,

$$\begin{aligned} p_{\beta-T} &\geq \inf_{x\in Q^*} C_M^{-1} \sum_{y\in Q^{**}} c_1 \beta^{-d/2} \exp\left\{-c_2 \frac{2(c_\eta \ell)^2}{\beta}\right\} \\ &= C_M^{-1} c_1 \ell^d \left(\frac{1}{\beta}\right)^{d/2} \exp\left\{-c_2 \frac{2(c_\eta \ell)^2}{\beta}\right\} \\ &\geq c_p. \end{aligned}$$

◀

B Standard large deviation results

▶ **Lemma 18** (Chernoff bound for Poisson). *Let P be a Poisson random variable with mean λ. Then, for any $0 < \epsilon < 1$,*

$$\mathbb{P}[P < (1-\epsilon)\lambda] < \exp\{-\lambda\epsilon^2/2\} \quad \textit{and} \quad \mathbb{P}[P > (1+\epsilon)\lambda] < \exp\{-\lambda\epsilon^2/4\}.$$

Flipping out with Many Flips: Hardness of Testing k-Monotonicity

Elena Grigorescu[1]
Purdue University, West Lafayette, IN, USA
https://www.cs.purdue.edu/homes/egrigore/
elena-g@purdue.edu

Akash Kumar[2]
Purdue University, West Lafayette, IN, USA
https://www.cs.purdue.edu/homes/akumar
akumar@purdue.edu

Karl Wimmer
Duquesne University, Pittsburgh, PA, USA
http://www.mathcs.duq.edu/~wimmer/

Abstract

A function $f : \{0,1\}^n \to \{0,1\}$ is said to be k-monotone if it flips between 0 and 1 at most k times on every ascending chain. Such functions represent a natural generalization of (1-)monotone functions, and have been recently studied in circuit complexity, PAC learning, and cryptography. Our work is part of a renewed focus in understanding testability of properties characterized by freeness of arbitrary order patterns as a generalization of monotonicity. Recently, Canonne et al. (ITCS 2017) initiate the study of k-monotone functions in the area of property testing, and Newman et al. (SODA 2017) study testability of families characterized by freeness from order patterns on real-valued functions over the line $[n]$ domain.

We study k-monotone functions in the more relaxed *parametrized property testing model*, introduced by Parnas et al. (JCSS, 72(6), 2006). In this process we resolve a problem left open in previous work. Specifically, our results include the following.

1. Testing 2-monotonicity on the hypercube non-adaptively with one-sided error requires an exponential in $\sqrt{n}$ number of queries. This behavior shows a stark contrast with testing (1-)monotonicity, which only needs $\tilde{O}(\sqrt{n})$ queries (Khot et al. (FOCS 2015)). Furthermore, even the apparently easier task of distinguishing 2-monotone functions from functions that are far from being $n^{.01}$-monotone also requires an exponential number of queries.
2. On the hypercube $[n]^d$ domain, there exists a testing algorithm that makes a constant number of queries and distinguishes functions that are k-monotone from functions that are far from being $O(kd^2)$-monotone. Such a dependency is likely necessary, given the lower bound above for the hypercube.

2012 ACM Subject Classification Theory of computation → Sketching and sampling, Theory of computation → Lower bounds and information complexity

Keywords and phrases Property Testing, Boolean Functions, k-Monotonicity, Lower Bounds

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.40

Acknowledgements We thank our collaborators Clément Canonne and Siyao Guo, who have gracefully refused to co-author this paper.

[1] Supported by NSF CCF-1649515
[2] Supported by NSF CCF-1649515 and NSF CCF-1618918

1 Introduction

Property testing [15, 54, 35] studies the complexity of deciding if a large object satisfies a property, or is far from satisfying the property, when the algorithm has only partial access to its input. Two prolific lines of research in the study of Boolean functions have recently seen ultimate results: on one hand, the study of families exhibiting algebraic symmetries (such as low-degree polynomials, and triangle-freeness), explicitly initiated in [41], and on the other hand, the study of functions with less symmetry, but whose values respect a monotone order relation, initiated in [34]. Extending the structural features of these classes of functions, in this work we view families of Boolean functions with less symmetry as interpolating from basic families of monotone functions to families characterized by freeness of more complex *order patterns*. This perspective is apparent in the recent works of Newman et al [51] and Canonne et al [17] (a superset of the authors) who propose the study of families exhibiting freeness from more general order patterns in the property testing setting. Here we study the property of being *k-monotone*, building towards understanding testability of freeness from arbitrary order patterns.

We first introduce some standard definitions. A *property* of Boolean functions from a discrete domain D to $\{0,1\}$ is a subset of $\{f\colon D \to \{0,1\}\}$. Given two functions $f, g\colon D \to \{0,1\}$, denote by $d(f,g)$ the (normalized) Hamming distance between them, i.e. $d(f,g) = \Pr_{x\sim D}[\, f(x) \neq g(x)\,]$. The distance of a function f to $\mathcal{P}$ is defined as $d(f,\mathcal{P}) = \min_{g\in\mathcal{P}} d(f,g)$. A *$q$-query tester* for $\mathcal{P}$ is a randomized algorithm $\mathcal{T}$ that takes as input $\varepsilon \in (0,1]$, and has query access to a function $f\colon D \to \{0,1\}$. After making at most $q(\varepsilon)$ queries, $\mathcal{T}$ distinguishes between the following two cases: i) if $f \in \mathcal{P}$, then $\mathcal{T}$ accepts, with probability 2/3; ii) if $d(f,\mathcal{P}) \geq \varepsilon$, then $\mathcal{T}$ rejects w.p. 2/3. If the algorithm only errs in the second case but accepts any function $f \in \mathcal{P}$ with probability 1, it is said to be *one-sided*; otherwise, it is said to be *two-sided*. Moreover, if the queries made to the function can only depend on the internal randomness of the algorithm, but not on the values obtained during previous queries, it is said to be *non-adaptive*; otherwise, it is *adaptive*. The maximum number of queries made to f in the worst case is the *query complexity* of the testing algorithm. When $d(f,\mathcal{P}) \geq \epsilon$, we say that f is *ϵ-far* from $\mathcal{P}$. We will frequently use "far" to denote "$\Omega(1)$-far".

In this work we focus on the property of being k-monotone, which we formalize next.

▶ **Definition 1** (k-monotonicity)**.** A function $f : D \to \{0,1\}$ is *k-monotone* if there do not exist $x_1 \prec x_2 \prec \ldots \prec x_{k+1}$ in D, such that $f(x_1) = 1$ and $f(x_i) \neq f(x_{i+1})$ for all $i \in [k]$. Equivalently, a Boolean k-monotone function is free of the pattern $\langle f(x_1), f(x_2), \ldots, f(x_{k+1})\rangle = \langle 1,0,1,0,\ldots,0,1\rangle$ [3]. Note that 1-monotone functions are exactly the functions that are monotone.

Testing monotonicity has drawn interest in the theoretical computer science community for almost two decades (e.g., [34, 28, 30, 33, 32, 4, 1, 39, 11, 16, 31, 13, 20, 19, 21, 7, 25, 24, 43, 5, 26]), especially due to the naturalness of the property, as well as its evasiveness to tight analysis.

The notion of k-monotonicity has been studied ever since the 50's in the context of circuit lower bounds [48]. Indeed, k-monotone functions correspond to functions computable by Boolean circuits with $\log k$ negation gates, and in particular monotone functions correspond to circuits with no negation gates. In proving lower bounds for circuits with few negation

[3] Note that $f(x_{k+1})$ could be 0 depending on the parity of k

gates, it has been apparent that the presence of even one negation gate leads either to failure of the common analysis techniques, or to failure of the expected results, as remarked by Jukna [40].

Interest in k-monotonicity has been recently rekindled from multiple angles, including PAC learning, circuit complexity, cryptography and Fourier analysis [53, 38, 37, 47, 27]. In these areas, as k increases, k-monotone functions are viewed as robust classes sitting between the very structured monotone functions, and general Boolean functions. The study of k-monotonicity in the property testing model was recently initiated in [17]. Here we continue this study and reveal a stark difference from testing monotonicity, while disproving a conjecture from [17]. Our study leads to open questions that might be relevant to monotonicity testing.

1.1 Testing 2-monotonicity on the hypercube

Recall that a function $f : \{0,1\}^n \to \{0,1\}$ is 2-monotone if it is free from $\langle f(x_1), f(x_2), f(x_3)\rangle = \langle 1,0,1\rangle$, for $x_1 \prec x_2 \prec x_3$. How much more difficult can one-sided testing of this pattern be than testing freeness from $\langle f(x_1), f(x_2)\rangle = \langle 1,0\rangle$ (i.e., monotonicity)?

The first proof of the $O(n)$-query one-sided non-adaptive tester for monotonicity from [34] relies on the following structural theorem: if f is far from monotone, then there are many *edges* that contain a violation to monotonicity (i.e., $x_1 \prec x_2$, with $\langle f(x_1), f(x_2)\rangle = \langle 1,0\rangle$, and there is no x_3 such that $x_1 \prec x_3 \prec x_2$). For k-monotonicity with $k \geq 2$, there is no such result: since all violations require at least 3 points, the violations can all be spread across many levels of the cube. For example, consider a totally symmetric *block function* $f(x)$ such that $f(x) = 1$ if $|x|$ is between $n/2 - 2\sqrt{n}$ and $n/2 - \sqrt{n}$, or between $n/2 + \sqrt{n}$ and $n/2 + 2\sqrt{n}$, and $f(x) = 0$ otherwise. This function is far from 2-monotone, yet any triple of points that witnesses this fact will contain a pair of points whose Hamming distance is at least $\sqrt{n}$.

This non-locality of a violation seems to be a reason why testing k-monotonicity (non-adaptively, with one-sided error) turns out to be very different than just testing monotonicity. Initial lower bounds from [17] only show a separation for $k \geq 3$. Indeed, they [17] show that testing k-monotonicity non-adaptively with one sided error requires $\Omega(n/k^2)^{k/4}$ queries, which for $k \geq 3$ beats the recent tester for monotonicity of $\tilde{O}(\sqrt{n})$ query complexity [19, 43]. However, no such separation was known for $k = 2$. Furthermore, they conjecture that k-monotonicity gets gradually more difficult to test as k increases, and that k-monotonicity should be testable with one-sided error with a budget of $\Theta(n^{O(k)})$ queries. In this work we show that this is not the case, and in fact even testing 2-monotonicity itself requires an exponential number of queries.

▶ **Theorem 2** (Rephrasing Corollary 12). *Testing 2-monotonicity non-adaptively with one-sided error requires $2^{\Omega(\sqrt{n})}$ queries.*

For comparison, [17] obtains a tester for 2-monotonicity with $2^{O(\sqrt{n}\cdot\log n)}$ queries, and [14] obtains a $2^{\Omega(\sqrt{n})}$ lower bound for PAC learning. Hence, testing 2-monotonicity is a natural property for which testing is essentially as hard as PAC learning.

While being contrary to the intuition that since monotonicity testing is easy, so should 2-monotonicity be, this result reinforces the theme discussed above in proving circuit complexity lower bounds. The class of 2-monotone functions is exactly the class of functions computable by a Boolean circuit with at most one negation gate [48, 14]. As in the circuit complexity world, allowing one negation gate significantly increases complexity.

A canonical test for one-sided testing

As alluded to above, in order to test k-monotonicity with one-sided error, a tester should only reject if it finds a *violation* in the form of a sequence $x_1 \prec x_2 \prec \ldots \prec x_{k+1}$ in $\{0,1\}^n$, such that $f(x_1) = 1$ and $f(x_i) \neq f(x_{i+1})$ for all $i \in [k]$. Hence, a canonical candidate tester suggested in [17] queries all points along a random chain and rejects only if it finds a violation.

▶ **Definition 3.** We define the (basic) *chain tester* to be the algorithm that picks a *uniformly random chain* $\mathbf{Z} = \langle 0^n \prec \mathbf{z}_1 \prec \mathbf{z}_2 \prec \cdots \mathbf{z}_{n-1} \prec 1^n \rangle$ of comparable points from $\{0,1\}^n$, and queries f at all these points. The chain tester rejects if $\mathbf{Z}$ reveals a violation to k-monotonicity, otherwise it accepts. We also sometimes denote by a chain tester an algorithm that picks multiple chains (possibly from a joint distribution).

All previous tests for monotonicity (e.g., [34, 20, 25, 43]) imply a chain tester incurring only a small polynomial blow-up in the query complexity.

As expected, a chain tester is indeed implied by any non-adaptive one-sided algorithm for k-monotonicity.

▶ **Theorem 4** (Corollary to Theorem 10)**.** *Any non-adaptive one-sided q-query tester for k-monotonicity over $\{0,1\}^n$ implies an $O\big(q^{k+1}n\big)$-query tester that queries points on a distribution over chains, and succeeds with constant probability. In particular, if p is the success probability of the basic chain tester, then $p = \Omega(1/q^{k+1})$.*

To prove our lower bounds, we create families for which the chain tester fails. We first remark that the chain tester does very well on the 'block functions' described above in Section 1.1 that are far from being 2-monotone. Indeed, every chain from 0^n to 1^n will uncover a violation to 2-monotonicity! In our constructions, we get around this by hiding such functions with "long" violations on a *small set of coordinates*, while still making sure it comprises a constant fraction of the cube. We show that a random chain is unlikely to visit enough of these coordinates to find a violation.

Proving that these functions are far from k-monotone amounts to understanding the structure of the violation hypergraph (i.e., the hypergraph whose vertices are elements of $\{0,1\}^n$, and whose edges are the tuples that witness a violation). The violation graph is to some extent the only handle we have on arguing structural properties of functions that are far from being k-monotone. A large matching (edges with disjoint sets of vertices) in this hypergraph implies that the function is far from k-monotone. Indeed, such families can be shown to have a large matching.

In fact, our results work in a larger context of parametrized testing, and our lower bounds hold even for apparently much easier tasks, as we describe next.

1.2 Parametrized monotonicity testing on the hypercube

The notion of k-monotonicity allows us to propose the natural problem of approximating the "monotonicity" of a function. For a concrete example, suppose we are promised that the unknown function f is either 2-monotone or far from, say, $n^{0.01}$-monotone. That is, either f changes value at most twice on every chain, or a constant fraction of points of $\{0,1\}^n$ need to be changed so that f changes value at most $n^{0.01}$ times on every chain. This promise problem can only require fewer queries, and intuitively, it should be much fewer.

However, we show that intuition is incorrect: even the apparently much easier task of distinguishing between functions that are 2-monotone and functions that are far from being $n^{.01}$-monotone requires an exponential number of queries.

▶ **Theorem 5** (Rephrasing Corollary 11). *Testing non-adaptively with one-sided error whether a function is 2-monotone or far from being $n^{0.01}$-monotone requires $2^{\Omega(n^{0.48})}$ queries.*

This is also the setting of parametrized property testing introduced by Parnas et al [52]. In this setting, a property is parametrized by an integer k and denoted $\mathcal{P} = \{\mathcal{P}_k\}_k$. A (k_1, k_2)-*tester* for the family $\mathcal{P}$ is a randomized algorithm which, on input a proximity parameter $\varepsilon \in (0, 1)$ and oracle access to an unknown function f, must accept if $f \in \mathcal{P}_{k_1}$, with probability at least $2/3$; and reject if $d(f, \mathcal{P}_{k_2}) > \varepsilon$, with probability at least $2/3$.

Hence, we are interested in $(k, g(k, n))$-testing for parametrized monotonicity, denoted $\{\mathcal{M}_k\}_k$. Following the notation just described, we will speak of property testers for (and query complexity lower bounds for) $(k, g(k, n))$-testing the property $\{\mathcal{M}_k\}_k$.

▶ **Theorem 6.** *Given $2 \leq k \leq g(k, n) = o(\sqrt{n})$, $(k, g(k, n))$-testing $\{\mathcal{M}_k\}_k$ (on the hypercube) non-adaptively with one-sided error requires $2^{\Omega\left(\frac{\sqrt{n}}{(k+1)(g(k,n)/k)^2}\right)}$ queries.*

Parametrized monotonicity testing may provide a new angle for approaching yet unanswered questions towards the goal of understanding the role of adaptivity in testing monotonicity. The current strong lower bounds for testing monotonicity adaptively [25, 24, 5] come polynomially close to the one-sided error upper bounds [43], yet the question of whether adaptive algorithms can beat the current lower bounds still remain open.

1.3 Parametrized monotonicity testing on the hypergrid

We next study parametrized monotonicity testing of Boolean functions over the hypergrid $[n]^d$ domain. For these domains [17] shows that testing k-monotonicity non-adaptively with two-sided error can be performed with a number of queries independent of n, but *exponential* in the dimension d. More explicitly, $q(n, d, \varepsilon, k) = \min\left(\tilde{O}\left(\frac{1}{\varepsilon^2}(5kd/\varepsilon)^d\right), 2^{\tilde{O}\left(k\sqrt{d}/\varepsilon^2\right)}\right)$.

Our algorithmic results show a contrast to the hypercube case. We obtain a tester with *constant* query complexity (independent of k, n, d), albeit trading off for the (k_1, k_2) parameters that the tester needs to distinguish between.

▶ **Theorem 7.** *There is a non-adaptive two-sided tester which performs* $\mathrm{poly}(1/\varepsilon)$ *queries and $(k, 2kd^2/\varepsilon)$-tests $\{\mathcal{M}_k\}_k$ on the hypergrid $[n]^d$.*

Two-sided versus one-sided testing for monotonicity on the hypercube is an unresolved problem. Thus, in light of our exponential bounds for the $\{0, 1\}^n$ domain, the dependence in d in the $g(n, d, k) = 2kd^2$ function appears necessary. Achieving a sublinear dependence on d while controlling the query complexity would require new ideas that also apply to the hypercube.

1.4 Related work and the larger perspective

As previously mentioned, [51] proposes studying more general order patterns for real-valued functions defined over the line domain $[n]$. They view a pattern of length k as a permutation $\pi : [k] \to [k]$. A function $f : [n] \to \mathbb{R}$ if π-free if there do not exist indices $i_1 < i_2 < \ldots < i_k$ such that $f(i_x) < f(i_y)$ whenever $\pi(x) < \pi(y)$. This is a more fine-grained notion than k-monotonicity; for example, $(2, 1)$-freeness describes monotonicity, and the intersection of $(2, 1, 3)$-freeness and $(3, 1, 2)$-freeness describes 2-monotonicity over the reals. The authors of [51] obtain several results for non-adaptive one-sided testing on the line, essentially distinguishing monotone and non-monotone patterns. They further raise the question of characterizing the testing complexity as a function of the structure of the pattern. Their

lower bounds are $\Omega(\sqrt{n})$; this is quite strong given that the domain size is n, and this bound is likely not tight for most patterns. Recently, [6] showed that for "most" permutations π testing π-freeness requires $\frac{n^{1-1/(k-\theta(1))}}{\varepsilon^{1/(k-\theta(1))}}$ queries. Transferring this question to our setting, we may ask:

▶ **Question 8.** *On the hypercube, do all order patterns (longer than some constant length) require* $\exp(\Omega(\sqrt{n}))$ *queries to test? Can this lower bound be strengthened to* $\exp(\Omega(n))$ *for most patterns? Can these patterns be characterized?*

The description of a function family as being free from a pattern is common in property testing of Boolean functions, especially in the study of families that exhibit 'affine-invariance'. In that line of work, the Boolean domain $\{0,1\}^n$ is viewed as a vector space over the field of two elements, and now one may perform algebraic operations over the domain. For instance, the property of being 'triangle-free' is defined as freeness from the pattern $\langle f(x), f(y), f(z)\rangle = \langle 1,1,1\rangle$ whenever $x+y+z=0$. An almost complete characterization of when such a property is testable with a constant number of queries has recently been obtained as a result of sustained interest in this quest [36, 46, 55, 8, 10, 9, 56, 12].

Monotonicity and k-monotonicity however exhibit less symmetric structure, but these families are still invariant under permutations of the variables. Invariance under permutations of the variables is also maintained by properties defined by freeness of order patterns. While this symmetry is not so strong as to imply constant query testers in general, it is however crucial in the design of algorithms for such properties. For example, we use this invariance in showing the reduction to a canonical tester in Theorem 4.

The notion of k-monotonicity can be easily extended to real-valued functions. In fact, this extension has already implied 'tolerant' monotonicity testers for families consisting of functions $f\colon [n]^d \to [0,1]$ [17]. Furthermore, [17] reveals connections between testing k-monotonicity and testing surface area [42, 2, 45, 50], as well as estimating support of distributions [18].

Besides the relevant extensive literature on monotonicity testing mentioned before, another recent direction that generalizes monotonicity is testing *unateness*. Namely, a unate function is monotone on each full chain, however, edge-disjoint chains may be monotone in different directions. Initiated in [34], query-efficient unateness testers were obtained in [34, 23, 44, 3], and lower bounds in [26, 3].

A trickle of recent work in cryptography focuses on understanding how many negation gates are needed to compute cryptographic primitives, such as one-way permutations, small bias generators, hard core bits, and extractors [38, 37, 47]. Very recently, motivated by connections to log rank conjecture and the sensitivity conjecture, [27] explore connections between the sparsity, sensitivity and alteration complexity of a boolean function (which is essentially identical to the notion of k-monotonicity).

Organization

We proceed with the proofs of our lower bounds for the hypercube domain in Section 2 and with our algorithmic results for the hypergrid in Section 3.

2 Lower Bounds over the Hypercube

We prove all of our results in this section. We first show in Theorem 9 that the basic chain tester detects a violation with negligibly small probability, and hence the $(k, g(k,n))$-problem is hard for chain testers.

▶ **Theorem 9.** *Given $2 \leq k \leq g(k,n) = o(\sqrt{n})$, there exist $C > 0$, and a collection $\mathcal{F}$ of Boolean functions, such that*

(i) *every $f \in \mathcal{F}$ is $\Omega(1)$-far from being $g(k,n)$-monotone, and*

(ii) *the probability that a uniformly random chain in $\{0,1\}^n$ detects a violation to k-monotonicity for f is $o\left(\exp\left(-C\frac{\sqrt{n}}{(g(k,n)/k)^2}\right)\right)$.*

We then show that any other non-adaptive, one-sided tester gives rise to a chain tester, with only a small blowup in the query complexity.

▶ **Theorem 10.** *Any non-adaptive one-sided q-query $(k, g(k,n))$-tester for Boolean monotonicity over $\{0,1\}^n$ implies an $O\left(q^{k+1}n\right)$-query tester that queries points on a distribution over chains, and succeeds with constant probability. In particular, if p is the success probability of the basic chain tester, then $p = \Omega(1/q^{k+1})$.*

Theorem 9 and Theorem 10 imply Theorem 6.

Instantiating k and $g(k,n)$ in Theorem 6, we obtain the following immediate corollaries.

▶ **Corollary 11.** *Any non-adaptive one-sided $(2, n^{0.01})$-tester for parametrized monotonicity requires $\exp(\Omega(n^{0.48}))$ queries.*

▶ **Corollary 12.** *Let $2 \leq k = o(\sqrt{n})$. Then any non-adaptive, one-sided tester for k-monotonicity requires $\Omega\left(\exp\left(\frac{\sqrt{n}}{k+1}\right)\right)$ queries.*

Note that the lower bound of Corollary 12 is $> \exp(\sqrt[4]{n})$ for $2 \leq k \leq \sqrt[4]{n}$. Using the previous lower bound of $\Omega\left(\left(\frac{n}{k^2}\right)^{k/4}\right)$ for any $2 \leq k = o(\sqrt{n})$ from [17], we obtain the following immediate consequence.

▶ **Corollary 13.** *Any non-adaptive, one sided tester for k-monotonicity, for $2 \leq k = o(\sqrt{n})$, requires $\Omega(\exp(\sqrt[4]{n}))$ queries.*

2.1 Proof of Theorem 9

Define the *weight* of an element x in $\{0,1\}^n$, denoted $|x|$, to be the number of non-zero entries of x.

We first recall some standard useful facts.

▶ **Fact 14.** *The maximum value of $\binom{n}{t}$ occurs when $t = \lfloor n/2 \rfloor$, and this maximum value is less than $2 \cdot 2^n/\sqrt{n}$.*

▶ **Fact 15.** *There exists a constant $C > 0$, such that for every $\varepsilon > 0$, the number of points of $\{0,1\}^n$ that with weight outside the* middle levels *$[\frac{n}{2} - \sqrt{n}\log\frac{C}{\varepsilon}, \frac{n}{2} + \sqrt{n}\log\frac{C}{\varepsilon}]$ is at most $\varepsilon 2^{n-1}$.*

In Section 2.1.1, we define our hard family, and show that every function in this family is indeed far from $g(k,n)$-monotonicity. In Section 2.1.2 we show that this family is hard for the chain tester.

The hard family hides instances of a *balanced blocks* function, which was previously used in [17] towards proving that testing k-monotonicity is at least as hard as testing monotonicity, even for an adaptive two-sided tester.

▶ **Definition 16** (Balanced Blocks function)**.** For every n and $\ell \leq o(\sqrt{n})$, let us partition the vertex set of the Hamming cube into ℓ blocks $B_1, B_2, \ldots, B_\ell$ where every block B_i consists of

all points in *consecutive levels* of the Hamming cube, such that all of the blocks have *roughly* the same size, i.e., for every $i \in \ell$, we have

$$\left(1 - \frac{\ell}{\sqrt{n}}\right)\frac{2^n}{\ell} \leq |B_i| \leq \left(1 + \frac{\ell}{\sqrt{n}}\right)\frac{2^n}{\ell}.$$ [4]

Then the corresponding *balanced blocks function with ℓ blocks*, denoted[5] $\mathrm{BB}(n,\ell)\colon \{0,1\}^n \to \{0,1\}$, is defined to be the blockwise constant function which takes value 1 on all of B_1 and whose value alternates on consecutive blocks.

Note that $\mathrm{BB}(n,\ell)$ is a totally symmetric function: it is unchanged under permutations of its inputs. Thus, we can partition $\{0,1,\ldots,n\}$ into ℓ intervals $I_1, I_2, \ldots, I_\ell$, such that I_i is the set of Hamming levels that B_i contains.

[17] shows that Balanced Blocks functions satisfy a useful property that we soon recall in Claim 18, after making an important definition.

▶ **Definition 17** (Violation hypergraph with respect to ℓ-monotonicity ([17])). Given a function $f\colon \{0,1\}^n \to \{0,1\}$, the *violation hypergraph of f* is $H_{\mathrm{viol}}(f) = (\{0,1\}^n, E(H_{\mathrm{viol}}))$ where $(x_1, x_2, \cdots, x_{\ell+1}) \in E(H_{\mathrm{viol}})$ if the ordered $(\ell+1)$-tuple $x_1 \prec x_2 \prec \ldots \prec x_{\ell+1}$ (which is a $(\ell+1)$-uniform hyperedge) forms a violation to ℓ-monotonicity in f. A collection M_h of pairwise disjoint $(\ell+1)$-uniform hyperedges of the violation hypergraph is said to form a *violated matching*.

▶ **Claim 18** ([17, Claim 3.8]). *Let $h \stackrel{\mathrm{def}}{=} \mathrm{BB}(n,\ell)$. Then h is $(\ell-1)$-monotone and not $(\ell-2)$-monotone. Furthermore, the violation graph of h with respect to $(\ell-2)$-monotonicity contains a violated matching of size at least $\frac{(1-o(1))2^n}{\ell}$, where every edge of the matching $y_1 \preceq \cdots \preceq y_{\ell-1}$ has $h(y_1) = 1$ and $h(y_i) \neq h(y_{i+1})$ for $1 \leq i < \ell - 1$.*

This machinery allows us to deduce the following.

▶ **Claim 19.** *Let $h \stackrel{\mathrm{def}}{=} \mathrm{BB}(n,3k)$. Then $d(h, \mathcal{M}_k) \geq \Omega(1)$* [6].

Proof of Claim 19. By Claim 18, $H_{\mathrm{viol}}(h)$ contains a matching M_h of $(3k-1)$-tuples of size $\frac{(1-o(1))2^n}{3k}$, and for every tuple in the matching $y_1 \preceq \cdots \preceq y_{3k-1}$ we have $h(y_1) = 1$ and $h(y_i) \neq h(y_{i+1})$ for $1 \leq i < 3k-1$. We see that any k-monotone function close to h must have at most k flips within any such tuple, by definition. It follows that any k-monotone function differs from h in at least $k-1$ vertices in every tuple of M_h. Thus, the Hamming distance between h and any k-monotone function is at least

$$\frac{(1-o(1))2^n}{3k} \cdot (k-1) \geq \frac{2^n}{5}.$$ ◀

We are now ready to describe the hard family.

2.1.1 The Hard Family

In what follows, let $s \stackrel{\mathrm{def}}{=} g(k,n)$. Also, let $r \stackrel{\mathrm{def}}{=} \frac{g(k,n)}{k}$. We now describe the hard instance for (k,s)-testing.

---

[4] The blocks are stacked successively. The block with index i+1 lies immediately above the one with index i.

[5] We will arbitrarily fix a function that satisfies these conditions.

[6] d(f,g) here denotes the Hamming distance between boolean functions f and g defined on the binary cube. Thus, 19 says that the function h is sufficiently far from being k-monotone.

Consider the partition of the set of indices in $[n]$ into two different *disjoint* sets, L and R, with sizes $|L| = n_L = n \cdot (1 - \frac{1}{625r^2})$ and $|R| = n_R = \frac{n}{625r^2}$, respectively.[7] We will write $z \in \{0,1\}^n$ as $z = (x, y)$, where $x \in \{0,1\}^{|L|}$ and $y \in \{0,1\}^{|R|}$. We define $\mathrm{MID}_L \stackrel{\text{def}}{=} \{i :| \, i - \frac{n_L}{2} \, | \leq \frac{\sqrt{n_L}}{100}\}$, to denote the set of "balanced" inputs restricted to the set of indices in L.

Assuming k is even[8], we define $f_{n_L} : \{0,1\}^n \to \{0,1\}$ by

$$f_{n_L}(x,y) \stackrel{\text{def}}{=} \begin{cases} 0, & \text{if } |x| \notin \mathrm{MID}_L \\ \mathrm{BB}(n_R, 3s)(y), & \text{otherwise i.e., } |x| \in \mathrm{MID}_L, \end{cases}$$

where $x \in \{0,1\}^{|L|}$ and $y \in \{0,1\}^{|R|}$.

For $x \in \{0,1\}^L$, let us denote by H_x the restriction of the hypercube $\{0,1\}^n$ to the points (x, y), with $y \in \{0,1\}^{|R|}$. Note that for $x \in \mathrm{MID}_L$, the restriction of the function to H_x is a copy of balanced blocks function on $n_R = n/625r^2$ variables with $3s$ blocks.

▶ **Claim 20.** *The function $f = f_{n_L}$ is $\Omega(1)$-far from being s-monotone.*

Proof. By Fact 15, picking $\varepsilon = \frac{1}{3}$, say, it follows that for a constant fraction of the points $x \in \{0,1\}^{n_L}$, the function f restricted to the cube H_x is a balanced blocks function on $3s$ blocks. By Claim 18, there is a matching of violations to $(3s-2)$-monotonicity within the violation graph on H_x, of size at least $\Omega(1) \cdot \frac{2^{n_R}}{3s}$. It follows that the there is a matching of violations to $(3s-2)$-monotonicity on the whole domain $\{0,1\}^n$ of size at least $\Omega(1)\frac{2^n}{3s}$. As in the proof of Claim 19, to produce a function that is s-monotone, one must change the value of f in at least s points of each matched hyper-edge. It follows that f is $\Omega(1)$-far from being s-monotone. ◀

Our hard family of functions is the orbit of the function f_{n_L} under all the permutations of the variables.

▶ **Definition 21.** The family $\mathcal{F}_{k,s}$, parameterized by k and s is defined as follows. Setting $n_L = (1 - \frac{k^2}{625s^2})n$, we define

$$\mathcal{F}_{k,s} \stackrel{\text{def}}{=} \{ \, f_{n_L} \circ \pi_\sigma \, : \, \sigma \in S_n \, \}$$

where $\pi_\sigma : \{0,1\}^n \to \{0,1\}^n$ is a permutation that sends the string $\{(a_1, a_2, \ldots, a_n)\}$ to $\{a_{\sigma(1)}, a_{\sigma(2)}, \ldots, a_\sigma(n)\}$ for a permutation $\sigma : [n] \to [n]$. We omit the parameters k and s if it is clear from the context.

We now observe that these functions are indeed far from being s-monotone. This follows since s-monotonicity is closed under permutation of the variables, and by Claim 20.

▶ **Claim 22.** *Every $f \in \mathcal{F}_{k,s}$, f is $\Omega(1)$-far from s-monotone.*

Therefore, we proved the distance property from Theorem 9.

[7] For the sake of presentation, we ignore integrality issues where possible.

[8] For odd k, the function is defined almost analogously – the only difference is that $f_{n_L}(x,y) = 1$ whenever $|x| > \frac{n_L}{2} + \frac{\sqrt{n_L}}{100}$. We make this assumption throughout the paper.

2.1.2 The Hard Family vs. the Chain Tester

Recalling that the basic chain tester picks a uniformly random chain in $\{0,1\}^n$, note that the distribution of the queries chosen by the chain tester is unchanged over permutations of the variables. Thus, it suffices to analyze the probability that the chain tester discovers a violation to k-monotonicity on f_{n_L}. We will show that this probability is very small if the quantity s/k is small.

▶ **Claim 23.** *There exists a constant $C > 0$, such that the probability that a random chain reveals a violation to k-monotonicity in f_{n_L} is at most* $\exp\left(-C\frac{k^2}{s^2}\sqrt{n}\right)$.

Let Z be a fixed chain $0^n = z_0 \prec z_1 \prec z_2 \prec \cdots \prec z_n = 1^n$ in $\{0,1\}^n$. Note that $f_{n_L}(z_i) = f_{n_L}(x_i, y_i) = 0$ if $|x_i| \notin \mathrm{MID}_L$. Thus, if there is a violation to k-monotonicity in Z, then it can be found among points in Z where $|x_i| \in \mathrm{MID}_L$. Thus, a chain can only exhibit a violation on points $z_i = (x_i, y_i)$ where $n_L/2 - \sqrt{n_L}/100 \leq |x_i| \leq n_L/2 + \sqrt{n_L}/100$. By definition, regardless of the exact structure of x_i in this interval, $f_{n_L}(x_i, y_i) = \mathrm{BB}(n_R, 3s)(y_i)$. Since BB is a totally symmetric function, to determine if Z exhibits a violation, it is enough to analyze the set

$$V(Z) \stackrel{\text{def}}{=} \{j : \text{there exists } z_i = (x_i, y_i) \in Z \text{ such that } |x_i| \in \mathrm{MID}_L \text{ and } |y_i| = j\}.$$

We remark that for every chain Z, $V(Z)$ is a set of consecutive integers.

▶ **Claim 24.** *Suppose Z contains a violation to k-monotonicity. Then $|V(Z)| \geq k\sqrt{n_R}/(16s)$.*

Proof. By Fact 14, the maximum value of $Pr_{\mathbf{y}\sim\{0,1\}^{n_R}}[|\mathbf{y}| = t]$ over values of t is $2/\sqrt{n_R}$. Since $BB(n_R, 3s)$ has $3s$ blocks, the number of consecutive levels of I_i in any block B_i must satisfy

$$(2/\sqrt{n_R})|I_i| \geq \frac{1}{3s}(1 - o(1)) \geq \frac{1}{4s},$$

so $|I_i| \geq \sqrt{n_R}/(8s)$. To see a violation to k-monotonicity, the chain Z must contain points from each Hamming level in $k-1$ complete blocks, so this requires $|V(Z)| \geq (k-1)\sqrt{n_R}/(8s) \geq k\sqrt{n_R}/(16s)$, as claimed. ◀

We will show that that $|V(\mathbf{Z})|$ reaching this value is very unlikely for a random chain $\mathbf{Z}$. Let $\mathbf{Z}$ be a random chain $0^n \prec \mathbf{z}_1 \prec \cdots \prec \mathbf{z}_{n-1} \prec 1^n$.

Proof of Claim 23. The proof follows from Claim 24 and the following claim.

▶ **Claim 25.** *Let $\mathbf{Z}$ be a random chain. Then*

$$\Pr[|V(\mathbf{Z})| \geq k\sqrt{n_R}/(20s)] \leq \exp\left(-\frac{0.00009}{r^2}\sqrt{n}\right).$$

Proof. Let j be the smallest index such that $\mathbf{z}_j = (\mathbf{x}_j, \mathbf{y}_j)$ satisfies $|\mathbf{x}_j| = n_L/2 - \sqrt{n_L}/100$. This is the index where the chain enters the region where it could find violations.

Let w be the largest index such that $\mathbf{z}_w = (\mathbf{x}_w, \mathbf{y}_w)$ satisfies $|\mathbf{x}_w| = n_L/2 + \sqrt{n_L}/100$. If $\mathbf{Z}$ contains a violation to k-monotonicity, then it must occur on the subchain $\mathbf{z}_{j-1} \prec \mathbf{z}_j \prec \cdots \prec \mathbf{z}_w \prec \mathbf{z}_{w+1}$. By construction, we have $f(\mathbf{z}_\ell) = 0$ if $\ell \leq j-1$ or $\ell \geq w+1$. Further, $V(\mathbf{Z}) = \{|\mathbf{y}_j|, |\mathbf{y}_{j+1}|, \ldots, |\mathbf{y}_{w-1}|, |\mathbf{y}_w|\}$, and $|V(\mathbf{Z})| = |\mathbf{y}_w| - |\mathbf{y}_j| + 1$. Thus, to prove the claim, it satisfies to analyze $|\mathbf{y}_w| - |\mathbf{y}_j|$. Note $w - j$ is exactly $\sqrt{n_L}/50 + |V(\mathbf{Z})| - 1$; this accounts for $\sqrt{n_L}/50$ flips of variables in L and $|V(\mathbf{Z})| - 1$ flips of variables in R. Informally,

we want to show that the ratio of the number of variables flipped in L to number of variables flipped in R is, with very high probability, too small for the chain tester to succeed in finding a violation to k-monotonicity.

To simplify our analysis, we will not work directly with w. Instead, define j as above, but consider $\mathbf{z}_{j'} = (\mathbf{x}_{j'}, \mathbf{y}_{j'})$, where $j' = j + \sqrt{n}/3$. We will show that, with high probability, $j' \geq w$, and $|\mathbf{y}_{j'}| - |\mathbf{y}_j|$ (and thus $|\mathbf{y}_w| - |\mathbf{y}_j|$) is small.

We claim that the value of $|\mathbf{y}_{j'}| - |\mathbf{y}_j|$ is a random variable with a (random) hypergeometric distribution. Indeed, to draw a random variable distributed as this difference, we construct the following experiment that simulates the behavior of a random chain with respect to the function f:

- The chain tester picks $\sqrt{n}/3$ coordinates from the $n - |\mathbf{z}_j|$ coordinates set to 0.
- $n_R - |\mathbf{y}_j|$ of these coordinates are "successes" for the chain tester, which correspond to flipping variables in R, and
- $n_L - |\mathbf{x}_j|$ of these coordinates are "failures" for the chain tester, which correspond to flipping variables in L.

Let $H(u, N, t, i)$ denote the probability of seeing exactly i successes in t independent samples, drawn uniformly and without replacement, from a population of N objects containing exactly u successes.

The chain tester is most likely to see successes in the above experiment if $|\mathbf{y}_j| = 0$; we will assume that this happens. As seen in the proof of Claim 24, in order to successfully reject f, the chain must witness at least $\frac{k\sqrt{n_R}}{16s}$ successes. Let $t = \frac{k\sqrt{n_R}}{20s}$.

Note that if the number of successes is $i < t$, then the number of failures is at least $\frac{\sqrt{n}}{3} - t \geq \frac{\sqrt{n}}{3} - \frac{\sqrt{n}}{500} > \frac{\sqrt{n_L}}{50}$, and so in this case we have $|V(\mathbf{Z})| < t$; this corresponds to the chain missing a complete balanced block. It follows that the proof reduces to upper bounding the quantity

$$\Pr[|V(\mathbf{Z})| \geq t] = \sum_{i \geq t} H(n_R, n_L/2 + \sqrt{n_L}/100 + n_R, \sqrt{n}/3, i).$$

We analyze the above quantity using a Chernoff bound for hypergeometric random variables, where $\mathbf{X} = |\mathbf{y}_{j'}| - |\mathbf{y}_j|$.

▶ **Theorem 26** (Theorem 1.17 in [29]). *Let $\mathbf{X}$ be a hypergeometrically distributed random variable. Then*

$$\Pr[\mathbf{X} \geq \frac{5}{4} \cdot \mathbb{E}[\mathbf{X}]] \leq \exp(-\mathbb{E}[\mathbf{X}]/48).$$

We use the following claim.

▶ **Claim 27.** $\frac{4}{15} \cdot t \leq \mathbb{E}[\mathbf{X}] \leq \frac{4}{5} \cdot t$.

Proof. Standard facts about the hypergeometric distribution imply that

$$\mathbb{E}[\mathbf{X}] = \frac{\sqrt{n}}{3} \cdot \frac{n_R}{n_L/2 + \sqrt{n_L}/100 + n_R}.$$

Recall that $r = s/k \geq 1$, $n_L = n(1 - 1/(625r^2)) > 2n/3$, $n_R = n/(625r^2)$, and $t = \frac{k\sqrt{n_R}}{20s} = \frac{\sqrt{n_R}}{20r} = \frac{\sqrt{n}}{500r^2}$. It follows that

$$n_L/2 + \sqrt{n_L}/100 + n_R > n_L/2 > n/3.$$

Therefore

$$\mathbb{E}[\mathbf{X}] < \frac{\sqrt{n}}{3} \cdot \frac{3n_R}{n} = \sqrt{n} \cdot \frac{1}{625r^2} = \frac{4}{5}\frac{\sqrt{n}}{500r^2} = \frac{4}{5} \cdot t.$$

Since $n_L/2 + \sqrt{n_L}/100 + n_R < n$,

$$\mathbb{E}[\mathbf{X}] > \frac{\sqrt{n}}{3} \cdot \frac{n_R}{n} = \frac{\sqrt{n}}{3} \cdot \frac{1}{625r^2} = \frac{4}{15}\frac{\sqrt{n}}{500r^2} = \frac{4}{15} \cdot t. \quad \blacktriangleleft$$

We can now conclude that

$$\begin{aligned}
\Pr[\mathbf{X} \geq t] &= \Pr[|V(\mathbf{Z})| \geq t] = \sum_{i \geq t} H(n_R, n_L/2 + \sqrt{n_L}/100, \sqrt{n}/3, i) \\
&= \Pr\left[X \geq \mathbb{E}[\mathbf{X}] \cdot \frac{t}{\mathbb{E}[\mathbf{X}]}\right] \\
&\leq \Pr\left[X \geq \mathbb{E}[\mathbf{X}] \cdot \frac{5}{4}\right] \\
&\leq \exp(-\mathbb{E}[\mathbf{X}]/48) \\
&\leq \exp(-t/180) = \exp\left(\frac{\sqrt{n}}{90000r^2}\right).
\end{aligned}$$

◀

◀

2.2 The reduction

We now prove Theorem 10. As mentioned, Theorem 10 and Theorem 9 immediately imply Corollary 6.

Proof of Theorem 10. We show that given a q-query non-adaptive, one-sided (k, s)-tester, one can obtain a $O\big(q^{k+1}n\big)$-query (k, s)-tester that only queries values on a distribution over random chains.

Let T be a q-query non-adaptive, one-sided (k, s)-monotonicity tester. Therefore, T accepts functions that are k-monotone, and rejects functions that are ε-far from being s-monotone with probability 2/3.

Define a tester T' that on input a function f does the following: it first gets the queries of T, then for each $(k+1)$-tuple $q_1 \prec q_2 \prec \cdots \prec q_{k+1}$, T' queries an entire uniformly random chain from 0^n to 1^n, conditioned on containing these $k+1$ points. Therefore, T' is also one-sided, makes $O(\binom{q}{k+1}n) = O(q^{k+1}n)$ queries, and its success probability is no less than [9] the success probability of T.

Define T'' that on input f picks a random permutation $\boldsymbol{\sigma}\colon [n] \to [n]$ and then applies the queries of T' to the function $f \circ \pi_{\boldsymbol{\sigma}}$ (where $\pi_{\boldsymbol{\sigma}}$ is defined as in Definition 21). This means that if T' queries q, T'' queries $\pi_{\boldsymbol{\sigma}}(q)$. Then T'' ignores what T' answers and only rejects if it finds a violation on any one of the chains.

Note that if f is k-monotone, then so is $f \circ \pi_{\boldsymbol{\sigma}}$, and if f is ε-far from being s-monotone, then so is $f \circ \pi_{\boldsymbol{\sigma}}$.

[9] We assume that every query made by T belongs to at least one $(k+1)$-tuple. Queries that do not are of no help to T, since these queries can not be part of a violation to k-monotonicity discovered by T, and we are assuming that T is non-adaptive and one-sided.

Therefore, T'' is one-sided, non-adaptive, and makes $O(q^{k+1}n)$ queries. Since T' is one-sided, it can only reject if it finds a $(k+1)$-tuple forming a violation to k-monotonicity. So if T' rejects, so does T'', and it follows that the success probability of T'' is at least the success probability of T', which is at least $2/3$.

We now claim that the queries of T'' are distributed as $O(\binom{q}{k+1})$ uniformly random chains. While the marginal distribution for each individual chain is the uniform distribution over chains, the joint distribution over these chains is not necessarily independent. Suppose T' queries points $q_1, q_2, \cdots q_{k+1}$ with $q_1 \prec q_2 \prec \cdots \prec q_{k+1}$. Then $\pi_{\boldsymbol{\sigma}}(q_i)$ is a uniformly random point on its weight level, and $\pi_{\boldsymbol{\sigma}}(q_1) \prec \pi_{\boldsymbol{\sigma}}(q_2) \prec \ldots \prec \pi_{\boldsymbol{\sigma}}(q_{k+1})$. It follows that a chain chosen uniformly at random conditioned on passing through these points is a uniformly random chain in $\{0,1\}^n$.

Let p be the success probability of the basic chain tester that picks a uniformly random chain in $\{0,1\}^n$ and rejects only if it finds a violation to k-monotonicity. Taking a union bound over the chains chosen by T'', the success probability of T'' is at most $p \cdot \binom{q}{k+1} \leq p \cdot q^{k+1}$. It follows that $p \cdot q^{k+1} \geq 2/3$, from which it easily follows that $p = \Omega(q^{1/(k+1)})$, concluding the proof. ◀

3 Upper Bounds over the Hypergrid

In this section we prove Theorem 7. In this section, we consider Boolean functions over the hypergrid $[n]^d$. For convenience, we will define $[n] = \{0, 1, 2, \ldots, n-1\}$. Assuming m divides n, we define $\mathcal{B}_{m,n} : [n]^d \to [m]^d$ be the mapping such that $\mathcal{B}_{m,n}(y)_i = \lfloor y_i/m \rfloor$ for $1 \leq i \leq d$. For $x \in [m]^d$, we define $\mathcal{B}^{-1}_{m,n}(x)$ to be the inverse image of x under $\mathcal{B}_{m,n}$. Specifically, $\mathcal{B}^{-1}_{m,n}(x)$ is the set of points of the form $m \cdot x + [n/m]^d$, using the standard definitions of scalar multiplication and coordinatewise addition. That is, $\mathcal{B}^{-1}_{m,n}$ is a "coset" of $[n/m]^d$ points in $[n]^d$. We will call these cosets *blocks*, and we will say that $h : [n]^d \to \{0,1\}$ is an m-block function if it is constant on $\mathcal{B}^{-1}_{m,n}(x)$ for every $x \in [m]^d$. For readability, we will often suppress the dependence on m and n.

▶ **Claim 28** (Claim 7.1, [17]). *Suppose $f : [n]^d \to \{0,1\}$ is k-monotone. Then there is an m-block function $h : [n]^d \to \{0,1\}$ such that $d(f,h) < kd/m$.*

▶ **Claim 29.** *Suppose $h : [n]^d \to \{0,1\}$ is an m-block function. Then h is $((m-1)d-1)$-monotone.*

Proof. Without loss of generality, we assume that $h(0^d) = 0$. Suppose for the sake of contradiction that h is an m-block function such that h contains a violation to $((m-1)d-1)$-monotonicity. Equivalently, there exists $y_0 \prec y_1 \prec \cdots \prec y_{(m-1)d-1}$ in $[n]^d$ such that $h(y_0) = 1$ and $h(y_i) \neq h(y_{i+1})$ for $0 \leq i \leq (m-1)d-2$. Since h is constant on each block, we have no two y_i's are in the same block. Thus the set $\{\mathcal{B}_{m,n}(y_i) : 0 \leq i \leq (m-1)d-1\}$ contains $(m-1)d$ distinct vectors in $[m]^d$ that can be totally ordered. This implies that 0^d and $(m-1)^d$ (this is a vector of d coordinates, all of which are $m-1$) are in this set. Clearly, 0^d is the "smallest" vector in this total order, and it follows from our definition of violation that $h(0^d) = 1$. This is a contradiction, since we assumed at the outset that $h(0^d) = 0$. Thus h does not contain a violation to $((m-1)d-1)$-monotonicity, and h is $((m-1)d-1)$-monotone. ◀

We define the m-block-coarsening of a function $f : [n]^d \to \{0,1\}$ to be the m-block function $h : [n]^d \to \{0,1\}$ such that $d(f,h)$ is as small as possible. We would like to use query access to f to get query access to h, but it is not guaranteed we can do this. Rather, for

each $x \in [m]^d$, we randomly select a set $\mathbf{S}_x$ of points in the block $\mathcal{B}^{-1}(x)$, where we choose $|\mathbf{S}_x|$ to be large enough such that

$$\left| \Pr_{\mathbf{z} \sim \mathbf{S}_{\mathcal{B}(y)}} [f(\mathbf{z}) = 0] - \Pr_{\mathbf{z} \sim \mathcal{B}^{-1}(\mathcal{B}(y))} [f(\mathbf{z}) = 0] \right| \leq 1/9,$$

with high probability. Our target query complexity is independent of m, so we can *not* necessarily query points from each $\mathbf{S}_x$; these points merely allow us to talk about a specific (randomly chosen) function. We denote by $\mathbf{h}' : [n]^d \to \{0, 1\}$ the m-block function such that

$$\mathbf{h}'(y) = \operatorname*{argmax}_{b \in \{0,1\}} \Pr_{\mathbf{z} \sim \mathbf{S}_{\mathcal{B}(y)}} [f(\mathbf{z}) = b],$$

breaking ties arbitrarily. In other words, $\mathbf{h}'$ can be thought of as the m-block function obtained by (randomized) "greedy" coarsening of f where we greedily select the majority bit (or the winning bit) over the random sample $\mathbf{S}_x$ to be the value h' over the corresponding block.

▶ **Claim 30.** *We have* $d(f, \mathbf{h}') \leq \frac{5}{4} d(f, h)$.

Proof. Let B be a block such that h and $\mathbf{h}'$ disagree. Then the wrong bit was estimated to be the majority value of f when restricted to B. By our construction of $\mathbf{h}'$, we must have

$$\left| \Pr_{\mathbf{y} \sim B} [f(\mathbf{y}) = h(\mathbf{y})] - \Pr_{\mathbf{y} \sim B} [f(\mathbf{y}) = \mathbf{h}'(\mathbf{y})] \right| \leq 1/9,$$

since exactly one of $h(\mathbf{y})$ and $\mathbf{h}'(\mathbf{y})$ is 0 and the other is 1. It follows that $\Pr_{\mathbf{y} \sim B}[f(\mathbf{y}) \neq h(\mathbf{y})] \geq 4/9$ and $\Pr_{\mathbf{y} \sim B}[f(\mathbf{y}) \neq \mathbf{h}'(\mathbf{y})] \leq 5/9$. Combining these inequalities, we get $\Pr_{\mathbf{y} \sim B}[f(\mathbf{y}) \neq \mathbf{h}'(\mathbf{y})] \leq \frac{5}{4} \Pr_{\mathbf{y} \sim B}[f(\mathbf{y}) \neq h(\mathbf{y})]$.

Clearly, if h and $\mathbf{h}'$ do not disagree on B, then the previous probabilities are equal and the inequality holds. Thus, for all blocks B, $\Pr_{\mathbf{y} \sim B}[f(\mathbf{y}) \neq \mathbf{h}'(\mathbf{y})] \leq \frac{5}{4} \Pr_{\mathbf{y} \sim B}[f(\mathbf{y}) \neq h(\mathbf{y})]$. The claim follows by taking the expected value of each side over a uniformly chosen block. ◀

Proof of Theorem 7. In our tester, we set $m = (2kd^2/\varepsilon + 1)/d + 1$, and the tester simply estimates $d(f, \mathbf{h}')$ to within $\pm\varepsilon/8$, which can be done with $\widetilde{O}(1/\varepsilon^2)$ queries. Note that query access to $\mathbf{h}'$ can be simulated because $\mathbf{h}'$ is the result of the randomized greedy coarsening described earlier. By Claim 28, if f is k-monotone, then there is an m-block function h such that

$$d(f, h) < kd/m = kd/((2kd^2/\varepsilon + 1)/d + 1) < kd^2/(2kd^2/\varepsilon) = \varepsilon/2,$$

and it follows that $d(f, \mathbf{h}') \leq \frac{5}{4} d(f, h) \leq \frac{5}{4}(\varepsilon/2) = 5\varepsilon/8$. By Claim 29, if f is far from $2kd^2/\varepsilon$-monotone, then it is ε-far from every m-block function, as for our setting of m, we have $((m-1)d - 1) = 2kd^2/\varepsilon$. It follows that $d(f, \mathbf{h}') \geq \varepsilon$. Thus, the tester correctly accepts if the estimate of $d(f, \mathbf{h}')$ is at most $3\varepsilon/4$ and correctly rejects if this estimate is at least $7\varepsilon/8$. ◀

References

1 Nir Ailon and Bernard Chazelle. Information theory in property testing and monotonicity testing in higher dimension. *Inf. Comput.*, 204(11):1704–1717, 2006.

2 Maria-Florina Balcan, Eric Blais, Avrim Blum, and Liu Yang. Active property testing. In *FOCS*, pages 21–30. IEEE Computer Society, 2012.

3 Roksana Baleshzar, Deeparnab Chakrabarty, Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and C. Seshadhri. Optimal unateness testers for real-valued functions: Adaptivity helps. *CoRR*, abs/1703.05199, 2017.

4 Tŭgkan Batu, Ronitt Rubinfeld, and Patrick White. Fast approximate PCPs for multidimensional bin-packing problems. *Inf. Comput.*, 196(1):42–56, 2005.

5 Aleksandrs Belovs and Eric Blais. A polynomial lower bound for testing monotonicity. In *STOC*, pages 1021–1032. ACM, 2016.

6 Omri Ben-Eliezer and Clément L. Canonne. Improved bounds for testing forbidden order patterns. In *SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, 2018.

7 Piotr Berman, Sofya Raskhodnikova, and Grigory Yaroslavtsev. L_p-testing. In *STOC*, pages 164–173. ACM, 2014.

8 Arnab Bhattacharyya, Victor Chen, Madhu Sudan, and Ning Xie. Testing linear-invariant non-linear properties. *Theory of Computing*, 7(1):75–99, 2011.

9 Arnab Bhattacharyya, Eldar Fischer, Hamed Hatami, Pooya Hatami, and Shachar Lovett. Every locally characterized affine-invariant property is testable. In *STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 429–436, 2013.

10 Arnab Bhattacharyya, Eldar Fischer, and Shachar Lovett. Testing low complexity affine-invariant properties. In *SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 1337–1355, 2013.

11 Arnab Bhattacharyya, Elena Grigorescu, Kyomin Jung, Sofya Raskhodnikova, and David P. Woodruff. Transitive-closure spanners. In *SODA*, pages 932–941. SIAM, 2009.

12 Arnab Bhattacharyya, Elena Grigorescu, and Asaf Shapira. A unified framework for testing linear-invariant properties. *Random Struct. Algorithms*, 46(2):232–260, 2015.

13 Eric Blais, Joshua Brody, and Kevin Matulef. Property testing lower bounds via communication complexity. *Computational Complexity*, 21(2):311–358, 2012.

14 Eric Blais, Clément L. Canonne, Igor Carboni Oliveira, Rocco A. Servedio, and Li-Yang Tan. Learning circuits with few negations. In *APPROX-RANDOM*, volume 40 of *LIPIcs*, pages 512–527. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015.

15 Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *J. Comput. Syst. Sci.*, 47(3):549–595, 1993.

16 Jop Briët, Sourav Chakraborty, David García-Soriano, and Arie Matsliah. Monotonicity testing and shortest-path routing on the cube. *Combinatorica*, 32(1):35–53, 2012.

17 Clément L. Canonne, Elena Grigorescu, Siyao Guo, Akash Kumar, and Karl Wimmer. Testing k-monotonicity. In *ITCS*, page (to appear), 2017.

18 Clément L. Canonne and Ronitt Rubinfeld. Testing probability distributions underlying aggregated data. In *ICALP (1)*, volume 8572 of *Lecture Notes in Computer Science*, pages 283–295. Springer, 2014.

19 Deeparnab Chakrabarty and C. Seshadhri. An $o(n)$ monotonicity tester for boolean functions over the hypercube. In *STOC*, pages 411–418. ACM, 2013. Journal version as [22].

20 Deeparnab Chakrabarty and C. Seshadhri. Optimal bounds for monotonicity and Lipschitz testing over hypercubes and hypergrids. In *STOC*, pages 419–428, 2013.

21 Deeparnab Chakrabarty and C. Seshadhri. An optimal lower bound for monotonicity testing over hypergrids. *Theory of Computing*, 10:453–464, 2014.

22 Deeparnab Chakrabarty and C. Seshadhri. An $o(n)$ Monotonicity Tester for Boolean Functions over the Hypercube. *SIAM J. Comput.*, 45(2):461–472, 2016.

23 Deeparnab Chakrabarty and C. Seshadhri. A $\widetilde{O}(n)$ non-adaptive tester for unateness. *CoRR*, 2016.

24 Xi Chen, Anindya De, Rocco A. Servedio, and Li-Yang Tan. Boolean function monotonicity testing requires (almost) $n^{1/2}$ non-adaptive queries. In *STOC*, pages 519–528. ACM, 2015.

25 Xi Chen, Rocco A. Servedio, and Li-Yang Tan. New algorithms and lower bounds for monotonicity testing. In *FOCS*, pages 286–295. IEEE Computer Society, 2014.

26 Xi Chen, Erik Waingarten, and Jinyu Xie. Beyond talagrand functions: New lower bounds for testing monotonicity and unateness. *CoRR*, abs/1702.06997, 2017.

27 Krishnamoorthy Dinesh and Jayalal Sarma. Alternation, sparsity and sensitivity : Bounds and exponential gaps. In *CALDAM 2018*. Springer, 2018.

28 Yevgeniy Dodis, Oded Goldreich, Eric Lehman, Sofya Raskhodnikova, Dana Ron, and Alex Samorodnitsky. Improved testing algorithms for monotonicity. In *RANDOM-APPROX*, volume 1671 of *Lecture Notes in Computer Science*, pages 97–108. Springer, 1999.

29 Benjamin Doerr. *Theory of Randomized Search Heuristics: Foundations and Recent Developments*. World Scientific Publishing Co., Inc., River Edge, NJ, USA, 2011.

30 Funda Ergün, Sampath Kannan, Ravi Kumar, Ronitt Rubinfeld, and Mahesh Viswanathan. Spot-checkers. *J. Comput. Syst. Sci.*, 60(3):717–751, 2000.

31 Shahar Fattal and Dana Ron. Approximating the distance to monotonicity in high dimensions. *ACM Trans. Algorithms*, 6(3), 2010.

32 Eldar Fischer. On the strength of comparisons in property testing. *Inf. Comput.*, 189(1):107–116, 2004.

33 Eldar Fischer, Eric Lehman, Ilan Newman, Sofya Raskhodnikova, Ronitt Rubinfeld, and Alex Samorodnitsky. Monotonicity testing over general poset domains. In *STOC 2002, Montréal, Québec, Canada*, pages 474–483, 2002.

34 Oded Goldreich, Shafi Goldwasser, Eric Lehman, Dana Ron, and Alex Samorodnitsky. Testing monotonicity. *Combinatorica*, 20(3):301–337, 2000.

35 Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *J. ACM*, 45(4):653–750, 1998.

36 Ben Green. A Szemerédi-type regularity lemma in abelian groups. *Geometric and Functional Analysis*, 15(2):340–376, 2005.

37 Siyao Guo and Ilan Komargodski. Negation-limited formulas. In *APPROX-RANDOM*, volume 40 of *LIPIcs*, pages 850–866. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015.

38 Siyao Guo, Tal Malkin, Igor Carboni Oliveira, and Alon Rosen. The power of negations in cryptography. In *TCC (1)*, volume 9014 of *Lecture Notes in Computer Science*, pages 36–65. Springer, 2015.

39 Shirley Halevy and Eyal Kushilevitz. Testing monotonicity over graph products. *Random Struct. Algorithms*, 33(1):44–67, 2008.

40 Stasys Jukna. *Boolean Function Complexity*. Springer, 2012.

41 Tali Kaufman and Madhu Sudan. Algebraic property testing: the role of invariance. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 403–412. ACM, 2008. doi:10.1145/1374376.1374434.

42 Michael J. Kearns and Dana Ron. Testing problems with sublearning sample complexity. *J. Comput. Syst. Sci.*, 61(3):428–456, 2000.

43 Subhash Khot, Dor Minzer, and Muli Safra. On monotonicity testing and Boolean isoperimetric type theorems. In *FOCS*, pages 52–58. IEEE Computer Society, 2015.

44 Subhash Khot and Igor Shinkar. An ~o(n) queries adaptive tester for unateness. In *APPROX/RANDOM 2016, Paris, France*, pages 37:1–37:7, 2016.

45 Pravesh Kothari, Amir Nayyeri, Ryan O'Donnell, and Chenggang Wu. Testing surface area. In *SODA*, pages 1204–1214. SIAM, 2014.

46 Daniel Král', Oriol Serra, and Lluís Vena. A removal lemma for systems of linear equations over finite fields. *Israel Journal of Mathematics*, pages 1–15, 2011. doi:10.1007/s11856-011-0080-y.

47 Chengyu Lin and Shengyu Zhang. Sensitivity conjecture and log-rank conjecture for functions with small alternating numbers. *CoRR*, abs/1602.06627, 2016.

48 A. A. Markov. On the inversion complexity of systems of functions. *Doklady Akademii Nauk SSSR*, 116:917–919, 1957. English translation in [49].

49 A. A. Markov. On the inversion complexity of a system of functions. *Journal of the ACM*, 5(4):331–334, October 1958.

50 Joe Neeman. Testing surface area with arbitrary accuracy. In *STOC*, pages 393–397. ACM, 2014.

51 Ilan Newman, Yuri Rabinovich, Deepak Rajendraprasad, and Christian Sohler. Testing for forbidden order patterns in an array. In *SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 1582–1597, 2017.

52 Michal Parnas, Dana Ron, and Ronitt Rubinfeld. Tolerant property testing and distance approximation. *J. Comput. Syst. Sci.*, 72(6):1012–1042, 2006. doi:10.1016/j.jcss.2006.03.002.

53 Benjamin Rossman. Correlation bounds against monotone NC^1. In *Conference on Computational Complexity* (CCC), 2015.

54 Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM-J-COMPUT*, 25(2):252–271, 1996.

55 Asaf Shapira. Green's conjecture and testing linear invariant properties. In *Property Testing - Current Research and Surveys [outgrow of a workshop at the Institute for Computer Science (ITCS) at Tsinghua University, January 2010]*, pages 355–358, 2010.

56 Yuichi Yoshida. A characterization of locally testable affine-invariant properties via decomposition theorems. In *STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 154–163, 2014.

How Long Can Optimal Locally Repairable Codes Be?

Venkatesan Guruswami[1]
Computer Science Department, Carnegie Mellon University, Pittsburgh, USA.
venkatg@cs.cmu.edu
https://orcid.org/0000-0001-7926-3396

Chaoping Xing
School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore.
xingcp@ntu.edu.sg
https://orcid.org/0000-0002-1257-1033

Chen Yuan[2]
Centrum Wiskunde & Informatica, Amsterdam, Netherlands.
Chen.Yuan@cwi.nl
https://orcid.org/0000-0002-3730-8397

Abstract

A locally repairable code (LRC) with locality r allows for the recovery of any erased codeword symbol using only r other codeword symbols. A Singleton-type bound dictates the best possible trade-off between the dimension and distance of LRCs — an LRC attaining this trade-off is deemed *optimal*. Such optimal LRCs have been constructed over alphabets growing linearly in the block length. Unlike the classical Singleton bound, however, it was not known if such a linear growth in the alphabet size is necessary, or for that matter even if the alphabet needs to grow at all with the block length. Indeed, for small code distances $3, 4$, arbitrarily long optimal LRCs were known over fixed alphabets.

Here, we prove that for distances $d \geqslant 5$, the code length n of an optimal LRC over an alphabet of size q must be at most roughly $O(dq^3)$. For the case $d = 5$, our upper bound is $O(q^2)$. We complement these bounds by showing the existence of optimal LRCs of length $\Omega_{d,r}(q^{1+1/\lfloor (d-3)/2 \rfloor})$ when $d \leqslant r + 2$. Our bounds match when $d = 5$, pinning down $n = \Theta(q^2)$ as the asymptotically largest length of an optimal LRC for this case.

2012 ACM Subject Classification Theory of computation → Error-correcting codes

Keywords and phrases Locally Repairable Code, Singleton Bound

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.41

Related Version A full version of the paper is available at `https://arxiv.org/abs/1807.01064`.

[1] This work was done when the author was visiting the School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore, and the Center of Mathematical Sciences and Applications, Harvard University. Research supported in part by NSF CCF-1563742.

[2] Most of this work was done when the author was with the School of Physical and Mathematical Science, Nanyang Technological University, Singapore. Research supported in part by ERC H2020 grant No.74079 (ALGSTRONGCRYPTO).

1 Introduction

Modern distributed storage systems have been transitioning to erasure coding based schemes with good storage efficiency in order to cope with the explosion in the amount of data stored online. Locally Repairable Codes (LRCs) have emerged as the codes of choice for many such scenarios and have been implemented in a number of large scale systems e.g., Microsoft Azure [10] and Hadoop [17].

A block code is called a locally repairable code (LRC) with locality r if every symbol in the encoding is a function of r other symbols. This enables recovery of any single erased symbol in a local fashion by downloading at most r other symbols. On the other hand, one would like the code to have a good minimum distance to enable recovery of many erasures in the worst-case. LRCs have been the subject of extensive study in recent years [8, 6, 16, 18, 11, 13, 5, 15, 19, 20, 3]. LRCs offer a good balance between very efficient erasure recovery in the typical case in distributed storage systems where a single node fails (or becomes temporarily unavailable due to maintenace or other causes), and still allowing recovery of the data from a larger number of erasures and thus safeguarding the data in more worst-case scenarios.

A Singleton-type bound for locally repairable codes relating its length n, dimension k, minimum distance d and locality r was first shown in the highly influential work [6]. It states that a linear locally repairable code C must obey[3]

$$d(C) \leqslant n - k - \left\lceil \frac{k}{r} \right\rceil + 2. \tag{1}$$

Note that any linear code of dimension k has locality at most k, so in the case when $r = k$ the above bound specializes to the classical Singleton bound $d \leqslant n - k + 1$, and in general it quantifies how much one must back off from this bound to accommodate locality.

A linear LRC that meets the bound (1) with equality is said to be an *optimal* LRC. This work concerns the trade-off between alphabet size and code length for linear codes that are optimal LRCs. Initially, the existence of such optimal LRCs and constructions were only known over fields that were exponentially large in the block length [9, 18].[4]

In a celebrated paper, Tamo and Barg [19] constructed clever subcodes of Reed-Solomon codes that yield a class of optimal locally repairable codes inheriting the field size $q \approx n$ of Reed-Solomon codes. This shows that one can have optimal LRCs with a field size similar to that of Maximum Distance Separable (MDS) codes which attain the classical Singleton bound $d = n - k + 1$.

One is thus tempted to make an analogy between optimal LRCs and MDS codes. The famous MDS conjecture says that there are no non-trivial (meaning, distance $d > 2$) MDS codes with length exceeding $q + 1$ where q is its alphabet size, except in two corner cases (q even and $k = 3$, or $k = q - 1$) where the length is at most $q + 2$. This conjecture was famously resolved in the case when q is prime by Ball [1].

For optimal LRCs, it was shown that an analogous strong conjecture does not hold [13] for almost every distance d — using elliptic curves, they gave LRCs length $q + 2\sqrt{q}$ (an earlier construction using rational function fields achieved length $q + 1$ [11]). A construction

[3] The bound in [6] was shown even for a weaker requirement of locality only for the information symbols, but we focus on the more general all-symbol locality.

[4] If locality is desired only for the information symbols, then it is easy to construct optimal LRCs over linear-sized fields using any MDS code via the “Pyramid" construction [9]. As we said, our focus is on LRCs with all-symbol locality which is more challenging to ensure.

of length $n \approx \frac{r+1}{r}q$ was given for small distances in [21]. Note that all these constructions have length that is at most $O(q)$.

The MDS conjecture makes a very precise statement about the maximum possible length of MDS codes. An asymptotic upper bound of $n = O(q)$ (in fact even $n \leqslant 2q$) is much easier to establish for MDS codes. Given this apparent parallel and the above-mentioned constructions which don't achieve code lengths exceeding $O(q)$, one might wonder if the Tamo-Barg result is asymptotically optimal, in the sense that optimal LRCs must have length at most $O(q)$. Rather surprisingly, it was not even known if n must be bounded as a function of q at all — that is, it was conceivable that one could have arbitrarily long optimal LRCs over an alphabet of fixed size! Indeed, Barg et al. [2] gave optimal LRCs using algebraic surfaces of length $n \approx q^2$ when the distance $d = 3$ and $r \leqslant 4$. This then inspired the discovery of optimal LRCs with *unbounded length* for $d = 3, 4$ via cyclic codes [14]. In Appendix A.1, we include a simple construction of arbitrarily optimal LRCs for $d = 3, 4$ over any fixed field size that satisfies $q \geqslant r + 1$.

Our Results. Given this state of knowledge, the natural question that arises is whether there is any upper bound at all on the length of optimal locally repairable codes (as a function of its alphabet size). In this paper, we answer this question affirmatively. In fact, we show that as soon as the distance $d \geqslant 5$, one cannot have unbounded length optimal LRCs (unlike the cases of $d = 3, 4$). Below is a statement of our upper bound on the code length of optimal LRCs. To the best of our knowledge, this is the first upper bound on the length of optimal LRCs.[5]

▶ **Theorem 1** (Upper bound on code length of LRCs). *Let $d \geqslant 5$, and let C be an optimal LRC with locality r (that meets the bound (1) with equality) of length $n \geqslant \Omega(dr^2)$ over an alphabet of size q. Then $n \leqslant O(dq^3)$ when d is not divisible by 4, and $n \leqslant O(dq^{3+4/(d-4)})$ when $4|d$.*

Our actual upper bound is a bit better when $d \equiv 1 \pmod 4$ and in particular yields $n \leqslant O(q^2)$ when $d = 5$. The technical condition that n is at least $\Omega(dr^2)$ arises in ensuring that the code consists of $n/(r+1)$ *disjoint* recovery groups of size $(r+1)$ each, that together ensure recoverability with locality r for every codeword symbols.

In our second result, we complement the above result on the limitation of LRCs with a construction of super-linear (in q) length for $d \leqslant r + 2$.

▶ **Theorem 2** (Construction of long LRCs). *For every r, d with $d \leqslant r + 2$, there exist optimal LRCs of length $n \geqslant \Omega_{d,r}(q^{1+1/\lfloor (d-3)/2 \rfloor})$.*[6]

Again, to the best of our knowledge, this is the first code achieving super linear length in q for $d \geqslant 5$. The previous best construction due to [21] achieved a length of $\frac{r+1}{r}q$ for $d \leqslant r + 1$

Organization of the paper. The paper is organized as follows. In Section 2, we provide some preliminaries on locally repairable codes. In Section 3, we prove an upper bound on

[5] Using the bound of Theorem 1 of [4], one can deduce an upper bound of $O(qr)$ on the *distance* of optimal LRCs.

[6] When $d = r + 2$, it turns out that one cannot achieve bound (1) with equality; so we get codes with $d = n - k - \lceil k/r \rceil + 1$ which is the optimal trade-off in this case. For $d \leqslant r + 1$ we attain (1) with equality.

the length of optimal LRCs. In Section 4, we present a construction of optimal LRC with super linear length in its alphabet size. Due to space restrictions, some of the proofs are omitted and can be found in the full version.

2 Preliminaries

$[n]$ stands for $\{1, \ldots, n\}$. The floor function and ceiling function of x are denoted by $\lfloor x \rfloor$ and $\lceil x \rceil$, respectively. An $[n, k, d]_q$ code is a linear code over the field of size q that has length n, dimension k, and distance d. We now define the local recoverability property of a code formally. We give this definition in general without assuming linearity, though we restrict our focus to linear codes in this paper.

▶ **Definition 3.** Let C be a q-ary block code of length n. For each $\alpha \in \mathbb{F}_q$ and $i \in \{1, 2, \cdots, n\}$, define $C(i, \alpha) := \{\mathbf{c} = (c_1, \ldots, c_n) \in C \ : \ c_i = \alpha\}$. For a subset $I \subseteq \{1, 2, \cdots, n\} \setminus \{i\}$, we denote by $C_I(i, \alpha)$ the projection of $C(i, \alpha)$ on I. For $i \in \{1, 2, \cdots, n\}$, a subset R of $\{1, 2, \ldots, n\}$ that contains i is a called a *recovery set* for i if $C_{I_i}(i, \alpha)$ and $C_{I_i}(i, \beta)$ are disjoint for any $\alpha \neq \beta$, where $I_i = R \setminus \{i\}$. Furthermore, C is called a locally recoverable code with locality r if, for every $i \in \{1, 2, \cdots, n\}$, there exists a recovery set R_i for i of size $r + 1$.

▶ Remark 4. The above definition of recovery sets is slightly different from that of recovery sets given in literature where i is excluded in the recovery set I_i. The reason why we include i in the recover set R_i of i is for convenience of proofs in this paper.

For linear codes, which are the focus of this paper, the following lemma establishes a connection between the locality and the dual code $C^\perp$. The proof is folklore.

▶ **Lemma 5.** *A subset R of $\{1, 2, \ldots, n\}$ is a recovery set at i of a linear code C over $\mathbb{F}_q$ if and only if there exists a codeword in $C^\perp$ whose support contains i and is a subset of R.*

For a q-ary $[n, k, d]$-linear LRC with locality r, the Singleton-type bound says

$$d \leqslant n - k - \left\lceil \frac{k}{r} \right\rceil + 2. \tag{2}$$

Like the classical Singleton bound, the Singleton-type bound (2) does not take into account the cardinality of the code alphabet q. Augmenting this result, a recent work [4] established a bound on the distance of locally repairable codes that depends on q, sometimes yielding better results. However, in this paper, we specifically refer as optimal LRC a linear code achieving the bound (2). We now rewrite this bound in a form that will be more convenient to us.

▶ **Lemma 6.** *Let n, k, d, r be positive integers with $(r + 1)|n$. If the Singleton-type bound (2) is achieved, then*

$$n - k = \frac{n}{r+1} + d - 2 - \left\lfloor \frac{d-2}{r+1} \right\rfloor. \tag{3}$$

▶ Remark 7. It turns out that the other direction of Lemma 6 is also true if $d - 2 \not\equiv r \pmod{r+1}$.

3 An Upper Bound on Code Lengths

In this section, we investigate the upper bound on the code lengths of optimal LRCs over a finite field $\mathbb{F}_q$. For simplicity, we assume that n is divisible by $r + 1$ throughout this section. However, in Remark 9 and 11, we extend our results to cover the cases when n is not divisible by $r + 1$.

3.1 Justifying the assumption of disjoint recovery sets

We first argue that a r-local LRC with block length n divisible by $r+1$ can be assumed, under modest conditions on the parameters, to contain $n/(r+1)$ disjoint recovery sets that each allow for recovery of $(r+1)$ codeword symbols. This structure will then be helpful to us in upper bounding the length of LRCs.

We remark that the structure theorem in [6] showed that the information symbols can be arranged into k/r disjoint groups each with a local parity check, under the assumption that $r|k$. However, we seek all-symbol's locality, and their argument does not directly apply.

▶ **Lemma 8.** *Let C be an $[n,k,d]_q$ linear optimal LRC with locality r. Then, there exist $\frac{n}{r+1}$ disjoint recovery sets, each of size $r+1$ provided that*

$$\frac{n}{r+1} \geqslant \left(d-2-\left\lfloor\frac{d-2}{r+1}\right\rfloor\right)(3r+2)+\left\lfloor\frac{d-2}{r+1}\right\rfloor+1. \tag{4}$$

▶ Remark 9. A similar result holds when n is not divisible by $r+1$. In this case, one can guarantee $\lceil\frac{n}{r+1}\rceil$ recovery sets that cover all the n codeword positions.

3.2 Proving the upper bound

In this subsection, we prove Theorem 1 (restated more formally below) that gives an upper bound on the length n of a LRC in terms of its alphabet size q. The parity check view of an LRC will be instrumental in our argument, in a manner similar to the bound obtained for maximally recoverable (MR) LRCs in [7]. We will make use of Lemma 8 and the classical Hamming upper bound on the size of codes as a function of minimum distance to derive our result.

▶ **Theorem 10.** *Let C be an optimal $[n,k,d]_q$-linear locally repairable codes of locality r with $(r+1)|n$ and parameters satisfying the inequality (4) given in Lemma 8. If $d \geqslant 5$ and $d \equiv a \pmod 4$ for some $1 \leqslant a \leqslant 4$, then*

$$n = \begin{cases} O(dq^{\frac{4(d-2)}{d-a}-1}) & \text{if } a=1,2, \\ O(dq^{\frac{4(d-3)}{d-a}-1}) & \text{if } a=3,4. \end{cases} \tag{5}$$

In particular, we have $n = O\left(dq^{3+\frac{4}{d-4}}\right)$. Furthermore, we have $n = O(q^2)$, $O(q^3)$, $O(q^3)$, $O(q^4)$, $O(q^{2.5})$ and $O(q^3)$ for $d = 5,6,7,8,9,$ and 10, respectively.

Proof. Again we let $n-k = \frac{n}{r+1}+h$ with $h = d-2-\lfloor\frac{d-2}{r+1}\rfloor \leqslant d-2$. By Theorem 8, we know that there exist $\ell := \frac{n}{r+1}$ codewords $\mathbf{c}_1,\ldots,\mathbf{c}_\ell$ of $C^\perp$ such that the supports $\mathrm{Supp}(\mathbf{c}_1),\ldots,\mathrm{Supp}(\mathbf{c}_\ell)$, each of size $r+1$, are pairwise disjoint. Put $R_i = \mathrm{Supp}(\mathbf{c}_i)$. By considering an equivalent code, we may assume that $R_i = \{(i-1)(r+1)+1,\ldots,i(r+1)\}$ for $i = 1,2,\ldots,\ell$ and the projection of $\mathbf{c}_i$ at R_i are equal to all-one vector $\mathbf{1}$ of length $r+1$.

The parity-check matrix H has the following form

$$H = \left(\begin{array}{c|c|ccc|c} \mathbf{1} & \mathbf{0} & \cdots & \cdots & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \cdots & \cdots & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \cdots & \cdots & \mathbf{1} \\ \hline \multicolumn{6}{c}{A} \end{array}\right), \tag{6}$$

where A is an $h \times n$ matrix over $\mathbb{F}_q$. The submatrix consisting of the first ℓ rows of H is a block diagonal matrix. Let $\mathbf{h}_{i,j}$ be the $(i(r+1)+j)$-th column of H, i.e.,

$$\mathbf{h}_{i,j} = (\underbrace{0,\ldots,0}_{i-1}, 1, \underbrace{0,\ldots,0}_{\ell-i}, \mathbf{v}_{i,j})^T \tag{7}$$

for some $\mathbf{v}_{i,j} \in \mathbb{F}_q^h$, where T stands for transpose.

Define

$$\mathbf{h}'_{i,j} := \mathbf{h}_{i,j} - \mathbf{h}_{i,r+1} = (\underbrace{0,\ldots,0}_{\ell}, \mathbf{v}_{i,j} - \mathbf{v}_{i,r+1})^T$$

for $i \in [\ell]$ and $j \in [r]$. We claim that any $\lfloor \frac{d-1}{2} \rfloor$ of $\mathbf{h}'_{1,1},\ldots,\mathbf{h}'_{\ell,r}$ are linearly independent. Indeed, for any $t := \lfloor \frac{d-1}{2} \rfloor$ vectors $\mathbf{h}'_{i_1,j_1},\ldots,\mathbf{h}'_{i_t,j_t}$ and scalars $\lambda_{i_1,j_1},\ldots,\lambda_{i_t,j_t} \in \mathbb{F}_q$ satisfying $\sum_{k=1}^t \lambda_{i_k,j_k}\mathbf{h}'_{i_k,j_k} = \mathbf{0}$, i.e., $\sum_{k=1}^t \lambda_{i_k,j_k}(\mathbf{h}_{i_k,j_k} - \mathbf{h}_{i_k,r+1}) = \mathbf{0}$, we have $\sum_{k=1}^t \lambda_{i_k,j_k}\mathbf{h}_{i_k,j_k} - \sum_{k=1}^t \lambda_{i_k,j_k}\mathbf{h}_{i_k,r+1} = \mathbf{0}$

Note that $\mathbf{h}_{i_1,j_1},\ldots,\mathbf{h}_{i_k,j_k}$ together with $\mathbf{h}_{i_1,r+1},\ldots,\mathbf{h}_{i_k,r+1}$ are at most $2t \leqslant d-1$ distinct columns of H. It follows that they are linearly independent and thus the coefficient $\lambda_{i_1,j_1},\ldots,\lambda_{i_t,j_t}$ must be all zero.

Moreover, we note that the first ℓ components of $\mathbf{h}'_{i,j}$ are all zero for $(i,j) \in [\ell] \times [r]$. We shorten the vector $\mathbf{h}'_{i,j}$ by puncturing its first ℓ coordinates. Denote by $\widetilde{\mathbf{h}}_{i,j}$ the shortened vectors. It is clear that any $\lfloor \frac{d-1}{2} \rfloor$ of $\widetilde{\mathbf{h}}_{1,1},\ldots,\widetilde{\mathbf{h}}_{\ell,r}$ are still linearly independent. Let H_2 be the matrix whose columns consists of $\widetilde{\mathbf{h}}_{i,j}$ for $i = 1,\ldots,\ell$ and $j = 1,\ldots,r$ and let C_2 be a linear code whose parity-check matrix is H_2. Then C_2 is a linear code with length $N := n - \ell = \frac{rn}{r+1}$, dimension at least $N-h$ and distance at least $\lfloor \frac{d-1}{2} \rfloor + 1$. We now apply the Hamming bound to $C_2 = \left[n-\ell, \geqslant n-\ell-h, \geqslant \lfloor \frac{d-1}{2} \rfloor + 1\right]$-linear code.

Let $d = 4d_1 + a$ for some $d_1 \geqslant 1$ and $1 \leqslant a \leqslant 4$.

Case 1. $a = 1$ or 2. In this case, we have $\lfloor \frac{d-1}{2} \rfloor + 1 = 2d_1 + 1$. Applying the Hamming bound to C_2 gives

$$q^{N-h} \leqslant \frac{q^N}{\sum_{i=1}^{d_1}\binom{N}{i}(q-1)^i} \leqslant \frac{q^N}{\binom{N}{d_1}(q-1)^{d_1}} \leqslant \frac{q^N}{(\frac{N}{d_1})^{d_1}(q-1)^{d_1}},$$

i.e., $\frac{rn}{r+1} = N \leqslant \frac{d_1}{q-1} \times q^{\frac{h}{d_1}} = \frac{d-a}{4(q-1)} \times q^{\frac{4h}{d-a}} \leqslant \frac{d-a}{4(q-1)} \times q^{\frac{4(d-2)}{d-a}}$. The last inequality follows from the fact that $h \leqslant d-2$.

Case 2. $a = 3$ or 4. In this case, we have $\lfloor \frac{d-1}{2} \rfloor + 1 = 2d_1 + 2$. Deleting the first coordinate of C_2 gives a q-ary $[N-1, N-h, \geqslant 2d_1+1]$-linear code. Applying the Hamming bound to $[N-1, N-h, \geqslant 2d_1+1]$ gives

$$q^{N-h} \leqslant \frac{q^{N-1}}{\sum_{i=1}^{d_1}\binom{N-1}{i}(q-1)^i} \leqslant \frac{q^{N-1}}{\binom{N-1}{d_1}(q-1)^{d_1}} \leqslant \frac{q^{N-1}}{(\frac{N-1}{d_1})^{d_1}(q-1)^{d_1}},$$

i.e., $\frac{rn}{r+1} - 1 = N - 1 \leqslant \frac{d_1}{q-1} \times q^{\frac{h-1}{d_1}} = \frac{d-a}{4(q-1)} \times q^{\frac{4(h-1)}{d-a}} \leqslant \frac{d-a}{4(q-1)} \times q^{\frac{4(d-3)}{d-a}}$. In conclusion, we have

$$n \leqslant \begin{cases} \frac{r+1}{r} \times \frac{d-a}{4(q-1)} \times q^{\frac{4(d-2)}{d-a}} & \text{if } a = 1, 2, \\ \frac{r+1}{r}\left(\frac{d-a}{4(q-1)} \times q^{\frac{4(d-3)}{d-a}} + 1\right) & \text{if } a = 3, 4. \end{cases}$$

The desired result follows. ◀

▶ Remark 11. Let us extend this result to the case n is not divisible by $r+1$. From Remark 9, we obtain $\lceil \frac{n}{r+1} \rceil$ recovery sets $R_1, \ldots, R_{\lceil \frac{n}{r+1} \rceil}$ covering all of the n indices. There are at most $(r+1)\lceil \frac{n}{r+1} \rceil - n \leqslant r$ indices that belong to more than 1 of these $\lceil \frac{n}{r+1} \rceil$ recovery sets. We first build the parity-check matrix H whose first $\lceil \frac{n}{r+1} \rceil$ rows are $\mathbf{c}_1, \ldots, \mathbf{c}_{\lceil \frac{n}{r+1} \rceil}$ where $\mathbf{c}_i$ corresponds to recovery set R_i. Then, we remove the columns from H whose indices belong to multiple recovery sets. After removing at most r columns, we apply the same argument to the resulting matrix. It is thus clear that the same result also holds for the case n is not divisible by $r+1$, with a small adjustment of r in the final upper bound on the code length.

▶ Remark 12. From our proof of Theorem 10, one might see why our argument is not applicable to the optimal LRC with distance less than 5. In our argument, the optimal LRC of distance d is reduced to a code of distance at least $\lfloor \frac{d+1}{2} \rfloor$ without locality. If $d \leqslant 4$, this reduced code might be the Hamming code. As we know, the length of Hamming code is independent of the alphabet size. On the other hand, there indeed exists unbounded length of optimal LRC of distance $d \leqslant 4$. Therefore, our argument reveals the inherent differences of optimal LRCS with distance less than 5 and above.

4 Construction of LRCs of super-linear length

To the best of our knowledge, all known constructions of optimal LRCs have block length $n \leqslant O(q)$ unless $d \leqslant 4$. Our upper bound in the preceding section implies that n must be upper bounded by (roughly) q^3. A natural question arises whether there exists optimal LRC with super linear length in q, e.g, $n = \Omega(q^{1+\varepsilon})$ and some constant $d > 4$. In this section we answer this question affirmatively, showing such codes for all $d \leqslant r+2$.

When $d = r+2$ and $r+1 | n$, the Singleton-type bound (1) can't be met [6, Corollary 10]. In this case, by an optimal LRC we mean a code attaining the trade-off $d = n - k - \lceil \frac{k}{r} \rceil + 1$. When n is not divisible by $r+1$, by shortening the code, it is still possible to obtain the optimal LRCs.

▶ **Theorem 13.** *Assume $d \leqslant r+2$ and $(r+1)|n$. There exist optimal LRCs of length $n = \Omega_{d,r}(q^{1+\frac{1}{\lfloor (d-3)/2 \rfloor}})$. In particular, one obtains the best possible length $n = O(q^2)$ for optimal LRC of minimum distance 5 if $r \geqslant 3$ and $(r+1)|n$.*

Proof. Let $n = \eta q^{1+1/\lfloor (d-3)/2 \rfloor}$ with some constant η that only depends on d and r, i.e., $\eta = \Omega_{d,r}(1)$. We will determine η later. It suffices to construct a matrix H and show that the code C derived from this parity-check matrix is an optimal LRC. Label and order the n coordinates with $(i,j) \in [\frac{n}{r+1}] \times [r+1]$, i.e., (i_1, j_1) precedes (i_2, j_2) if $i_1 < i_2$ or $i_1 = i_2$ and $j_1 < j_2$. Let $H = (\mathbf{h}_{i,j})_{(i,j) \in [\frac{n}{r+1}] \times [r+1]}$ where $\mathbf{h}_{i,j} \in \mathbb{F}_q^{n-k}$. That means H consists of the columns $\mathbf{h}_{i,j}$ for $(i,j) \in [\frac{n}{r+1}] \times [r+1]$. We start from $\mathbf{h}_{1,1}$ and determine the value of $\mathbf{h}_{i,j}$ column by column in the above order. In each step, we make sure that the new column $\mathbf{h}_{i,j}$ together with any $d-2$ columns preceding the (i,j)-th column are linearly independent. Meanwhile, the matrix H holds the same form[7] as the matrix in (6). If we can achieve both of the conditions, we are done. Define $\frac{n}{r+1}$ blocks $B_1, \ldots, B_{\frac{n}{r+1}}$ such that $B_i = \{\mathbf{h}_{i,1}, \ldots, \mathbf{h}_{i,r+1}\}$. That means we partition the n columns into $\frac{n}{r+1}$ disjoint blocks. Algorithm 1 gives the iterative method to compute the columns $\mathbf{h}_{i,j}$'s.

[7] The same form is referred to that their distributions of non-zero entry in upper half matrix (matrix lying above A) are the same, i.e., entries of value 1 and 0 in this upper half matrix represents the nonzero and zero entries, respectively.

Algorithm 1

- For $i = 1, \ldots, \frac{n}{r+1}$, and $j = 1, \ldots, r+1$, do the following operation.
  - Find $\mathbf{v} \in \mathbb{F}_q^{n-k}$ of form (7)[8] such that $\mathbf{v}$ is linearly independent of any subset of at most $(d-2)$ columns $\mathbf{h}_{i,j}$ chosen before this step.
  - Let $\mathbf{v}$ be the (i,j)-th column of H, i.e., $\mathbf{h}_{i,j} = \mathbf{v}$.

We justify Algorithm 1 by showing that there always exists such $\mathbf{h}_{i,j}$ for any $(i,j) \in [\frac{n}{r+1}] \times [r+1]$. Assume that we arrive at the (a,b)-th column. If $b = 1$, the construction is trivial. Let $\mathbf{h}_{a,b}$ be a column vector such that the first $\frac{n}{r+1}$ components except i-th component are zero. Obviously, it matches the form of Equation 7. The linearly independence is also trivial since the i-th component of all the columns $\mathbf{h}_{i,j}$ for $i < a$ is 0. Otherwise, to simplify our discussion, we assume that the first $d-2$ columns are already found. Since any $d-2$ columns prior to the (a,b)-th column are already linearly independent by our algorithm, it suffices to show that $\mathbf{h}_{a,b}$ is linearly independent from these $d-2$ columns. To achieve this, we need to check all possible combinations of these $d-2$ columns. Assume that these $d-2$ columns are chosen exactly from t blocks. Obviously, block B_a must be selected. Otherwise, the same reason for $b = 1$ implies that $\mathbf{h}_{a,b}$ is linearly independent of these $d-2$ columns. Without loss of generality, we assume that these t blocks are $B_1, \ldots, B_{t-1}$ and B_a and there are i_j columns picked from block B_j. Then, the submatrix H_1 consisting of these $d-2$ columns has the following form:

$$H_1 = \left(\begin{array}{c|c|c|c|c} \mathbf{x}_1 & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{x}_2 & \cdots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{x}_{t-1} & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{x}_a \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \hline \multicolumn{5}{c}{A_1} \end{array}\right)$$

where $\mathbf{x}_i \in \mathbb{F}_q^{i_j}$ and A_1 is a $(d-2) \times (d-2)$ matrix. If any B_j, $j = 1, 2, \ldots, t-1$, contains only one column, then that column is linearly independent of the rest of the $d-3$ columns and $\mathbf{h}_{a,b}$, and therefore can be removed from consideration. Thus we may assume that there are at least two columns chosen in each block except block B_a. Thus, t is at most $\lfloor \frac{d-1}{2} \rfloor$. Recall that our goal is to ensure that $\mathbf{h}_{a,b}$ is linearly independent of these at most $d-2$ columns in total chosen from the blocks $B_1, \ldots, B_{t-1}$. Given the t blocks and the $d-2$ columns chosen from them, we count the number of bad $\mathbf{h}_{a,b}$ which are linear combinations of these $d-2$ columns. If the number of such linear combinations is smaller than the size of the whole space of possible choices of $\mathbf{h}_{a,b}$, we are done. To achieve this, we need to determine the maximal subspace V spanned by these $d-2$ columns such that all the vectors in V has the same form as the vector $\mathbf{h}_{a,b}$, i.e., the first $\frac{n}{r+1}$ components except a-th component are zero.

For block B_j with $j \neq a$, by the expression of matrix H_1, the i_j columns of B_j created a $i_j - 1$-dimensional subspace where the first $\frac{n}{r+1}$ components of all the vectors are 0. That

[8] Only the i-th component out of the first $\frac{n}{r+1}$ components is nonzero.

means, block B_j for $j \neq a$ contributes $i_j - 1$ linearly independent vectors to the maximal subspace V. For block B_a, it contributes at most i_a linearly independent vectors to the maximal subspace V. It follows that the dimension of V is at most $\sum_{i=1}^{t-1}(i_j-1)+i_a = d-1-t$. This implies that there are at most q^{d-1-t} $\mathbf{h}_{a,b}$s lying in the space spanned by these $d-2$ columns.

It remains to count the number of distinct $d-2$ column sets. Note that B_a is always selected. Thus, we only have at most $\binom{a-1}{t-1} \leqslant \binom{\frac{n}{r+1}}{t-1} \leqslant (\frac{n}{r+1})^{t-1}$ combinations of these t blocks. After fixing these t blocks, there are at most $(t(r+1))^{d-2}$ ways to pick $d-2$ columns from these t blocks due to the fact that these t blocks contain only $t(r+1)$ columns. In total, there are at most $(t(r+1))^{d-2}(\frac{n}{r+1})^{t-1}$ ways to pick $d-2$ columns that precede the (i,j)-th column. Each combination contributes to at most q^{d-1-t} bad $\mathbf{h}_{a,b}$. Thus, the number of bad $\mathbf{h}_{a,b}$ are upper bounded by

$$\begin{aligned}\sum_{t=1}^{\lfloor\frac{d-1}{2}\rfloor}\big(t(r+1)\big)^{d-2}\left(\frac{n}{r+1}\right)^{t-1}q^{d-1-t} &\leqslant \left(\frac{q(d-1)(r+1)}{2}\right)^{d-2}\sum_{t=1}^{\lfloor\frac{d-1}{2}\rfloor}\left(\frac{n}{q(r+1)}\right)^{t-1}\\ &\leqslant \left(\frac{q(d-1)(r+1)}{2}\right)^{d-2}\left(\frac{d-1}{2}\right)\left(\frac{n}{q(r+1)}\right)^{\lfloor\frac{d-3}{2}\rfloor}.\end{aligned}$$

The first inequality is due to $t \leqslant \frac{d-1}{2}$ and the last inequality is due to $n > q(r+1)$. Plug $n = \eta q^{1+\frac{1}{\lfloor(d-3)/2\rfloor}}$ into the formula. This number is upper bounded by $q^{d-1}(d-1)^{d-1}(r+1)^{(d-2)/2}\eta^{\lfloor(d-3)/2\rfloor}$; by picking η small enough as a function of d, r we can ensure this quantity is at most $q^{d-1}/2$.

On the other hand, according to Algorithm 1, the a-th component of $\mathbf{h}_{a,b}$ should be nonzero. Moreover, the first $\frac{n}{r+1}$ components except a-th component are all zero. That means, the whole space of $\mathbf{h}_{a,b}$ is of size $q^{d-1} - q^{d-2} > \frac{1}{2}q^{d-1}$. Thus, there always exists $\mathbf{h}_{a,b}$ satisfying our algorithm's requirement.

We are almost done. Let C be the code whose parity-check matrix is H. It is clear that C has locality r. Since any $d-1$ columns of H are linearly independent, C has minimum distance $d(C)$ at least d. Because H has $\frac{n}{r+1}+d-2$ rows, the dimension of C is $k(C) \geqslant n - \frac{n}{r+1} - (d-2) = \frac{rn}{r+1} - (d-2)$. This implies

$$\frac{k(C)}{r} \geqslant \frac{n}{r+1} - \frac{d-2}{r}$$

We divide it into two case.

- If $d-2 < r$, the condition $r+1|n$ implies $\left\lceil\frac{k(C)}{r}\right\rceil \geqslant \frac{n}{r+1}$ and thus $k(C)+\left\lceil\frac{k(C)}{r}\right\rceil \geqslant n-d+2$. It follows that

  $$d(C) \geqslant d \geqslant n - k(C) - \left\lceil\frac{k(C)}{r}\right\rceil + 2.$$

  Thus, C is an optimal LRC. We are done.
- If $d-2 = r$, the condition $r+1|n$ implies $\left\lceil\frac{k(C)}{r}\right\rceil \geqslant \frac{n}{r+1} - 1$ and thus $k(C) + \left\lceil\frac{k(C)}{r}\right\rceil \geqslant n-d+1$. It follows that

  $$d(C) \geqslant d \geqslant n - k(C) - \left\lceil\frac{k(C)}{r}\right\rceil + 1.$$

  C is still an optimal LRC because there does not exist LRC reaching the Singleton-type bound. ◀

Next, we extend this theorem to the case n is not divisible by $r+1$ and $d \leqslant r+2$.

▶ **Corollary 14.** *Assume* $n \equiv a \pmod{r+1}$ *and* $a > d-1$. *There exists optimal LRC of length* $n = \Omega_{d,r}(q^{1+\frac{1}{\lfloor (d-3)/2 \rfloor}})$. *In particular, one obtains the best possible length* $n = O(q^2)$ *for optimal LRC of minimum distance* 5 *if* $n \pmod{r+1} > 4$.

It was shown in [12, Theorem III.3] that under the assumption that C has $\lceil \frac{n}{r+1} \rceil$ disjoint recovery sets, a linear code C with length $n = a \bmod r+1$, $a \neq 0, 1$ and dimension either $k \bmod r \geqslant a$ or $r|k$ must obey that $d \leqslant n - k - \lceil \frac{k}{r} \rceil + 1$.

With the help of Theorem III.3 in [12], we can extend the result in Corollary 14 to cover the case $a \leqslant d-1$ and $a \neq 1$.

▶ **Corollary 15.** *Assume* $n \equiv a \pmod{r+1}$ *and* $a \neq 1$. *There exists optimal LRC of length* $n = \Omega_{d,r}(q^{1+\frac{1}{\lfloor (d-3)/2 \rfloor}})$.

References

1 S. Ball. On large subsets of a finite vector space in which every subset of basis size is a basis. *J. Eur*, 14:733–748, October 2012.

2 A. Barg, K. Haymaker, E. Howe, G. Matthews, and A. Várilly-Alvarado. Locally recoverable codes from algebraic curves and surfaces. In E. W. Howe, K. E. Lauter, and J. L. Walker, editors, *Algebraic Geometry for Coding Theory and Cryptography*, pages 95–126. s, Springer, 2017.

3 A. Barg, I. Tamo, and S. Vlăduţ. Locally recoverable codes on algebraic curves. *IEEE Trans. Inform.Theory*, 63:4928–4939, 2017.

4 V. Cadambe and A. Mazumda. Bounds on the size of locally recoverable codes. *IEEE Trans. Inform.Theory*, 61:5787–5794, 2015.

5 M. Forbes and S. Yekhanin. On the locality of codeword symbols in non-linear codes. *Discrete Mathematics*, 324(6):78–84, 2014.

6 P. Gopalan, C. Huang, H. Simitci, and S. Yekhanin. On the locality of codeword symbols. *IEEE Trans. Inform.Theory*, 58:6925–6934, 2012.

7 S. Gopi, V. Guruswami, and S. Yekhanin. On maximally recoverable local reconstruction codes. *Electronic Colloquium on Computational Complexity*, 24::183, 2017.

8 J. Han and L. A. Lastras-Montano. Reliable memories with subline accesses. In *Proc. IEEE Internat. Sympos. Inform. Theory*, pages 2531–2535, 2007.

9 C. Huang, M. Chen, and J. Li. Pyramid codes: Flexible schemes to trade space for access efficiency in reliable data storage systems. In *Sixth IEEE International Symposium on Network Computing and Applications*, pages 79–86, 2007.

10 C. Huang, H. Simitci, Y. Xu, Ogus A., B. Calder, P. Gopalan, J. Li, and S. Yekhanin. Erasure coding in windows azure storage. *In USENIX Annual Technical Conference (ATC)*, pages 15–26, 2012.

11 L. Jin, L. Ma, and Xing C. Construction of optimal locally repairable codes via automorphism groups of rational function fields. URL: `https://arxiv.org/abs/1710.09638`.

12 O. Kolosov, A. Barg, I. Tamo, and G. Yadgar. Optimal lrc codes for all lengths $n \leqslant q$. URL: `https://arxiv.org/pdf/1802.00157`.

13 X. Li, L. Ma, and C. Xing. *Optimal locally repairable codes via elliptic curves*. To appear in IEEE Trans. Inf. Theory, 2017. URL: `https://arxiv.org/abs/1712.03744`.

14 Y. Luo, C. Xing, and C. Yuan. *Optimal locally repairable codes of distance* 3 *and* 4 *via cyclic codes*. To appear in IEEE Trans. Inf. Theory, 2018. `arXiv:1801.03623`.

15 D. S. Papailiopoulos and A. G. Dimakis. Locally repairable codes. *IEEE Trans. Inform.Theory*, 60:5843–5855, 2014.

16 N. Prakash, G. M. Kamath, V. Lalitha, and P. V. Kumar. Optimal linear codes with a local-error-correction property. In *Proc. 2012 IEEE Int. Symp. Inform. Theory*, pages 2776–2780, 2012.

17 M. Sathiamoorthy, M. Asteris, D. S. Papailiopoulos, A. G. Dimakis, R. Vadali, S. Chen, and D. Borthakur. XORing elephants: novel erasure codes for big data. *Proceedings of VLDB Endowment (PVLDB)*, pages 325–336, 2013.

18 N. Silberstein, A. S. Rawat, O. O. Koyluoglu, and S. Vichwanath. Optimal locally repairable codes via rank-matric codes. In *Proc. IEEE Int. Symp. Inf. Theory*, pages 1819–1823, 2013.

19 I. Tamo and A. Barg. A family of optimal locally recoverable codes. *IEEE Trans. Inform. Theory*, 60:4661–4676, 2014.

20 I. Tamo, D. S. Papailiopoulos, and A. G. Dimakis. Optimal locally repairable codes and connections to matroid theory. *IEEE Trans. Inform. Theory*, 62:6661–6671, 2016.

21 Z. Zhang, J. Xu, and M. Liu. Constructions of optimal locally repairable codes over small fields. *SCIENTIA SINICA Mathematica*, 47(11):1607–1614, 2017.

A Appendix

A.1 Unbounded length LRCs for distances 3 and 4

▶ **Theorem 16.** *Assume that $d = 3, 4$, $d - 2 \leqslant r$ and $r + 1 | n$, there exist optimal LRCs of arbitrarily lengths as long as $q \geqslant r + 1$.*

Proof. For $d = 3, 4$, $d - 2 \leqslant r$ and $r + 1 | n$, the Singleton-type bound implies that $n - k = \frac{n}{r+1} + d - 2$. Since $q \geqslant r + 1$, we let A be a $(d - 2) \times (r + 1)$ Vandermonde matrix over $\mathbb{F}_q$ such that

$$A_1 = \begin{pmatrix} \mathbf{1} \\ A \end{pmatrix}$$

is a $(d - 1) \times (r + 1)$ Vandermonde matrix. Define $(\frac{n}{r+1} + d - 2) \times n$ matrix

$$H = \left(\begin{array}{c|c|c|c} \mathbf{1} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{1} \\ A & A & \cdots & A \end{array}\right)$$

Where $\mathbf{1}$ and $\mathbf{0}$ are all-1 and all-0 vectors in $\mathbb{F}_q^{r+1}$, respectively. We partition the columns of H into $\frac{n}{r+1}$ blocks $B_1, \ldots, B_{\frac{n}{r+1}}$ such that $\mathbf{h} \in B_i$ if its i-th component is non-zero. From the expression of matrix H, it is clear that each column belongs to exactly one block and the columns in distinct blocks are linearly independent. Moreover, any $d - 1$ columns in the same block are linearly independent due to the property of Vandermonde matrix A_1. Next we show that any $d - 1$ columns of H are linearly independent. It suffices to verify this claim for the case $d = 4$. To see this, we pick any three columns $\mathbf{h}_i, \mathbf{h}_j, \mathbf{h}_t$ from H. Nothing needs to prove if these three columns belong to the same block. We assume that they belong to at least two blocks. Without loss of generality, $\mathbf{h}_t$ is in a block that does not contain $\mathbf{h}_i$ and $\mathbf{h}_j$. From above observation, we see that $\mathbf{h}_t$ is linearly independent from $\mathbf{h}_i$ and $\mathbf{h}_j$. It is clear $\mathbf{h}_i$ and $\mathbf{h}_j$ are linearly independent no matter whether they belong to the same block or different blocks. Thus, any 3 columns of H are linearly independent. Let C be the linear code whose parity-check matrix is H. It is clear that C has length n, dimension $k(C) \geqslant n - \frac{n}{r+1} - (d - 2) = \frac{rn}{(r+1)} - (d - 2)$, distance $d(C) \geqslant d$ and locality r. The condition $d - 2 \leqslant r$ leads to $\lceil \frac{k(C)}{r} \rceil \geqslant \frac{n}{r+1}$ and thus $k(C) + \lceil \frac{k(C)}{r} \rceil \geqslant n - (d - 2)$. The desired result follows since $d(C) \geqslant d \geqslant n - k(C) - \lceil \frac{k(C)}{r} \rceil + 2$. ◀

On Minrank and Forbidden Subgraphs

Ishay Haviv
School of Computer Science, The Academic College of Tel Aviv-Yaffo, Tel Aviv 61083, Israel

Abstract

The *minrank* over a field $\mathbb{F}$ of a graph G on the vertex set $\{1, 2, \ldots, n\}$ is the minimum possible rank of a matrix $M \in \mathbb{F}^{n \times n}$ such that $M_{i,i} \neq 0$ for every i, and $M_{i,j} = 0$ for every distinct non-adjacent vertices i and j in G. For an integer n, a graph H, and a field $\mathbb{F}$, let $g(n, H, \mathbb{F})$ denote the maximum possible minrank over $\mathbb{F}$ of an n-vertex graph whose complement contains no copy of H. In this paper we study this quantity for various graphs H and fields $\mathbb{F}$. For finite fields, we prove by a probabilistic argument a general lower bound on $g(n, H, \mathbb{F})$, which yields a nearly tight bound of $\Omega(\sqrt{n}/\log n)$ for the triangle $H = K_3$. For the real field, we prove by an explicit construction that for every non-bipartite graph H, $g(n, H, \mathbb{R}) \geq n^{\delta}$ for some $\delta = \delta(H) > 0$. As a by-product of this construction, we disprove a conjecture of Codenotti, Pudlák, and Resta. The results are motivated by questions in information theory, circuit complexity, and geometry.

2012 ACM Subject Classification Mathematics of computing → Information theory

Keywords and phrases Minrank, Forbidden subgraphs, Shannon capacity, Circuit Complexity

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.42

Acknowledgements We are grateful to Alexander Golovnev and Pavel Pudlák for useful discussions and to the anonymous referees for their valuable suggestions.

1 Introduction

An $n \times n$ matrix M over a field $\mathbb{F}$ is said to *represent* a digraph $G = (V, E)$ with vertex set $V = \{1, 2, \ldots, n\}$ if $M_{i,i} \neq 0$ for every i, and $M_{i,j} = 0$ for every distinct i, j such that $(i, j) \notin E$. The *minrank* of G over $\mathbb{F}$, denoted $\mathrm{minrk}_{\mathbb{F}}(G)$, is the minimum possible rank of a matrix $M \in \mathbb{F}^{n \times n}$ representing G. The definition is naturally extended to (undirected) graphs by replacing every edge with two oppositely directed edges. It is easy to see that for every graph G the minrank parameter is sandwiched between the independence number and the clique cover number, that is, $\alpha(G) \leq \mathrm{minrk}_{\mathbb{F}}(G) \leq \chi(\overline{G})$. For example, $\mathrm{minrk}_{\mathbb{F}}(K_n) = 1$ and $\mathrm{minrk}_{\mathbb{F}}(\overline{K_n}) = n$ for every field $\mathbb{F}$. The minrank parameter was introduced by Haemers in 1979 [16], and since then has attracted a significant attention motivated by its various applications in information theory and in theoretical computer science (see, e.g., [17, 7, 32, 26, 25, 19, 10]).

In this work we address the extremal behavior of the minrank parameter of n-vertex graphs whose complements are free of a fixed forbidden subgraph. For two graphs G and H, we say that G is *H-free* if G contains no subgraph, induced or not, isomorphic to H. For an integer n, a graph H, and a field $\mathbb{F}$, let $g(n, H, \mathbb{F})$ denote the maximum of $\mathrm{minrk}_{\mathbb{F}}(G)$ taken over all n-vertex graphs G whose complement $\overline{G}$ is H-free. Our purpose is to study the quantity $g(n, H, \mathbb{F})$ where H and $\mathbb{F}$ are fixed and n is growing.

1.1 Our Contribution

We provide bounds on $g(n, H, \mathbb{F})$ for various graph families and fields. We start with a simple upper bound for a forest H.

▶ **Proposition 1.** *For every integer n, a field $\mathbb{F}$, and a nontrivial forest H on h vertices,*

$$g(n, H, \mathbb{F}) \leq h - 1.$$

Equality holds whenever H is a tree and $n \geq h - 1$.

We next provide a general lower bound on $g(n, H, \mathbb{F})$ for a graph H and a finite field $\mathbb{F}$. To state it, we need the following notation. For a graph H with $h \geq 3$ vertices and $f \geq 3$ edges define $\gamma(H) = \frac{h-2}{f-1}$ and $\gamma_0(H) = \min_{H'} \gamma(H')$, where the minimum is taken over all subgraphs H' of H with at least 3 edges.

▶ **Theorem 2.** *For every graph H with at least 3 edges there exists $c = c(H) > 0$ such that for every integer n and a finite field $\mathbb{F}$,*

$$g(n, H, \mathbb{F}) \geq c \cdot \frac{n^{1-\gamma_0(H)}}{\log(n \cdot |\mathbb{F}|)}.$$

Note that for every finite field $\mathbb{F}$, the quantity $g(n, H, \mathbb{F})$ grows with n if and only if H is not a forest. Indeed, if H is a forest then $g(n, H, \mathbb{F})$ is bounded by some constant by Proposition 1, whereas otherwise H satisfies $\gamma_0(H) < 1$ and thus, by Theorem 2, $g(n, H, \mathbb{F}) \geq \Omega(n^{\delta})$ for some $\delta = \delta(H) > 0$. Note further that for the case $H = K_3$, which is motivated by a question in information theory (see Section 1.2), Theorem 2 implies that

$$g(n, K_3, \mathbb{F}) \geq \Omega\Big(\frac{\sqrt{n}}{\log n}\Big) \tag{1}$$

for every fixed finite field $\mathbb{F}$. This is tight up to a $\sqrt{\log n}$ multiplicative term (see Proposition 13).

Theorem 2 is proved by a probabilistic argument based on the Lovász Local Lemma [13]. The proof involves an approach of Spencer [29] to lower bounds on off-diagonal Ramsey numbers and a technique of Golovnev, Regev, and Weinstein [15] for estimating the minrank of random graphs.

As our final result, we show that for every non-bipartite graph H there are H-free graphs with low minrank over the real field $\mathbb{R}$.

▶ **Theorem 3.** *For every non-bipartite graph H there exists $\delta = \delta(H) > 0$ such that for every sufficiently large integer n, there exists an n-vertex H-free graph G such that* $\mathrm{minrk}_{\mathbb{R}}(G) \leq n^{1-\delta}$.

This theorem is proved by an explicit construction from the family of generalized Kneser graphs, whose minrank was recently studied in [18]. It is known that every n-vertex graph G satisfies

$$\mathrm{minrk}_{\mathbb{F}}(G) \cdot \mathrm{minrk}_{\mathbb{F}}(\overline{G}) \geq n \tag{2}$$

for every field $\mathbb{F}$ (see, e.g., [23, Remark 2.2]). This combined with the graphs given in Theorem 3 implies the following (explicit) lower bound on $g(n, H, \mathbb{R})$ for non-bipartite graphs H.

▶ **Corollary 4.** *For every non-bipartite graph H there exists $\delta = \delta(H) > 0$ such that for every sufficiently large integer n, $g(n, H, \mathbb{R}) \geq n^{\delta}$.*

As another application of Theorem 3, we disprove a conjecture of Codenotti, Pudlák, and Resta [11] motivated by Valiant's approach to circuit lower bounds [31] (see Section 1.2).

1.2 Applications

The study of the quantity $g(n, H, \mathbb{F})$ is motivated by questions in information theory, circuit complexity, and geometry. We gather here several applications of our results.

Shannon Capacity

For an integer k and a graph G on the vertex set V, let G^k denote the graph on the vertex set V^k in which two distinct vertices $(u_1, \ldots, u_k)$ and $(v_1, \ldots, v_k)$ are adjacent if for every $1 \leq i \leq k$ it holds that u_i and v_i are either equal or adjacent in G. The Shannon capacity of a graph G, introduced by Shannon in 1956 [28], is defined as the limit $c(G) = \lim_{k\to\infty} (\alpha(G^k))^{1/k}$. This graph parameter is motivated by information theory, as it measures the zero-error capacity of a noisy communication channel represented by G. An upper bound on $c(G)$, known as the Lovász ϑ-function, was introduced in [22], where it was used to show that the Shannon capacity of the cycle on 5 vertices satisfies $c(C_5) = \sqrt{5}$, whereas its independence number is 2. Haemers introduced the minrank parameter in [16, 17] and showed that it forms another upper bound on $c(G)$ and that for certain graphs it is tighter than the ϑ-function. In general, computing the Shannon capacity of a graph seems to be a very difficult task, and its exact value is not known even for small graphs such as the cycle on 7 vertices.

The question of determining the largest possible Shannon capacity of a graph with a given independence number is widely open. In fact, it is not even known if the Shannon capacity of a graph with independence number 2 can be arbitrarily large [3]. Interestingly, Erdös, McEliece, and Taylor [14] have shown that this question is closely related to determining an appropriate multicolored Ramsey number, whose study in [33] implies that there exists a graph G with $\alpha(G) = 2$ and $c(G) > 3.199$. A related question, originally asked by Lovász, is that of determining the maximum possible ϑ-function of an n-vertex graph with independence number 2. This maximum is known to be $\Theta(n^{1/3})$, where the upper bound was proved by Kashin and Konyagin [20, 21], and the lower bound was proved by Alon [2] via an explicit construction. Here we consider the analogue question of determining the maximum possible minrank, over any fixed finite field $\mathbb{F}$, of an n-vertex graph with independence number 2. Since the latter is precisely $g(n, K_3, \mathbb{F})$, our bound in (1) implies that the minrank parameter is weaker than the ϑ-function with respect to the general upper bounds that they provide on the Shannon capacity of n-vertex graphs with independence number 2.

The Odd Alternating Cycle Conjecture

In 1977, Valiant [31] proposed the matrix rigidity approach for proving superlinear circuit lower bounds, a major challenge in the area of circuit complexity. Roughly speaking, the rigidity of a matrix $M \in \mathbb{F}^{n\times n}$ for a constant $\varepsilon > 0$ is the minimum number of entries that one has to change in M in order to reduce its rank over $\mathbb{F}$ to at most $\varepsilon \cdot n$. Valiant showed in [31] that matrices with large rigidity can be used to obtain superlinear lower bounds on the size of logarithmic depth arithmetic circuits computing linear transformations. With this motivation, Codenotti, Pudlák, and Resta [11] raised in the late nineties the Odd Alternating Cycle Conjecture stated below, and proved that it implies, if true, that certain explicit circulant matrices have superlinear rigidity. By an alternating odd cycle we refer to a digraph which forms a cycle when the orientation of the edges is ignored, and such that the orientation of the edges alternates with one exception.

▶ **Conjecture 5** (The Odd Alternating Cycle Conjecture [11]). *For every field $\mathbb{F}$ there exist $\varepsilon > 0$ and an odd integer ℓ such that every n-vertex digraph G with $\text{minrk}_{\mathbb{F}}(G) \leq \varepsilon \cdot n$ contains an alternating cycle of length ℓ.*

Codenotti et al. [11] proved that the statement of Conjecture 5 does not hold for $\ell = 3$ over any field $\mathbb{F}$. Specifically, they provided an explicit construction of n-vertex digraphs G, free of alternating triangles, with $\text{minrk}_{\mathbb{F}}(G) \leq O(n^{2/3})$ for every field $\mathbb{F}$. For the undirected case, which is of more interest to us, a construction of [11] implies that there are n-vertex triangle-free graphs G such that $\text{minrk}_{\mathbb{F}}(G) \leq O(n^{3/4})$ for every field $\mathbb{F}$ (see [8, Section 4.2] for a related construction over the binary field as well as for an application of such graphs from the area of index coding). Note that this yields, by (2), that $g(n, K_3, \mathbb{F}) \geq \Omega(n^{1/4})$. In contrast, for the real field and the cycle on 4 vertices, it was shown in [11] that every n-vertex C_4-free graph G satisfies $\text{minrk}_{\mathbb{R}}(G) > \frac{n}{6}$. Yet, the question whether every n-vertex digraph with sublinear minrank contains an alternating cycle of odd length $\ell \geq 5$ was left open in [11] for every field. Our Theorem 3 implies that for every odd ℓ there are (undirected) C_ℓ-free graphs G with sublinear $\text{minrk}_{\mathbb{R}}(G)$, and in particular disproves Conjecture 5 for the real field $\mathbb{R}$.

Nearly Orthogonal Systems of Vectors

A system of nonzero vectors in $\mathbb{R}^m$ is said to be nearly orthogonal if any set of three vectors of the system contains an orthogonal pair. It was proved by Rosenfeld [27] that every such system has size at most $2m$. An equivalent way to state this, is that every n-vertex graph represented by a real positive semidefinite matrix of rank smaller than $\frac{n}{2}$ contains a triangle. Note that the positive semidefiniteness assumption is essential in this result, as follows from the aforementioned construction of [11] of n-vertex triangle-free graphs G with $\text{minrk}_{\mathbb{R}}(G) \leq O(n^{3/4})$.

A related question was posed by Pudlák in [24]. He proved there that for some $\varepsilon > 0$, every n-vertex graph represented by a real positive semidefinite matrix of rank at most $\varepsilon \cdot n$ contains a cycle of length 5. Pudlák asked whether the assumption that the matrix is positive semidefinite can be omitted. Our Theorem 3 applied to $H = C_5$ implies that there are C_5-free graphs G with sublinear $\text{minrk}_{\mathbb{R}}(G)$, and thus answers this question in the negative.

1.3 Outline

The rest of the paper is organized as follows. In Section 2 we present the simple proof of Proposition 1. In Section 3 we provide some background on sparse-base matrices from [15] and then prove Theorem 2. In the final Section 4, we prove Theorem 3.

2 Forests

In this section we prove Proposition 1. We use an argument from one of the proofs in [5].

Proof of Proposition 1. Fix a nontrivial h-vertex forest H and a field $\mathbb{F}$. It suffices to consider the case where H is a tree, as otherwise H is a subgraph of some h-vertex tree H', and since every H-free graph is also H'-free, we have $g(n, H, \mathbb{F}) \leq g(n, H', \mathbb{F})$.

Our goal is to show that every n-vertex graph G whose complement $\overline{G}$ is H-free satisfies $\text{minrk}_{\mathbb{F}}(G) \leq h - 1$. Let G be such a graph. We claim that $\overline{G}$ is $(h-2)$-degenerate, that is, every subgraph of $\overline{G}$ contains a vertex of degree at most $h - 2$. Indeed, otherwise $\overline{G}$ has a subgraph G' all of whose degrees are at least $h - 1$, and one can find a copy of H in

G' as follows: First identify an arbitrary vertex of G' with an arbitrary vertex of H, and then iteratively identify a vertex of G' with a leaf added to the being constructed copy of the tree H. The process succeeds since H has h vertices and every vertex of G' has degree at least $h-1$. As is well known, the fact that $\overline{G}$ is $(h-2)$-degenerate implies that $\overline{G}$ is $(h-1)$-colorable, so we get that $\mathrm{minrk}_{\mathbb{F}}(G) \leq \chi(\overline{G}) \leq h-1$, as required.

We finally observe that the bound is tight whenever H is a tree and $n \geq h-1$. Indeed, let G be the n-vertex complete $\lceil \frac{n}{h-1} \rceil$-partite graph, that has $h-1$ vertices in each of its parts, except possibly one of them. Its complement $\overline{G}$ is a disjoint union of cliques, each of size at most $h-1$, and is thus H-free. Since $\alpha(G) = \chi(\overline{G}) = h-1$, it follows that $\mathrm{minrk}_{\mathbb{F}}(G) = h-1$ for every field $\mathbb{F}$, completing the proof. ◀

3 A General Lower Bound on $g(n, H, \mathbb{F})$

In this section we prove Theorem 2 and discuss its tightness for $H = K_3$. We start with some needed preparations.

3.1 Lovász Local Lemma

The Lovász Local Lemma [13] stated below is a powerful probabilistic tool in Combinatorics (see, e.g., [6, Chapter 5]). We denote by $[N]$ the set of integers from 1 to N.

▶ **Lemma 6.** *[Lovász Local Lemma [13]] Let $A_1, \ldots, A_N$ be events in an arbitrary probability space. A digraph $D = (V, E)$ on the vertex set $V = [N]$ is called a dependency digraph for the events $A_1, \ldots, A_N$ if for every $i \in [N]$, the event A_i is mutually independent of the events A_j with $j \neq i$ and $(i, j) \notin E$. Suppose that $D = (V, E)$ is a dependency digraph for the above events and suppose that there are real numbers $x_1, \ldots, x_N \in [0, 1)$ such that*

$$\Pr[A_i] \leq x_i \cdot \prod_{(i,j) \in E} (1 - x_j)$$

for all $i \in [N]$. Then, with positive probability no event A_i holds.

3.2 Sparse-base Matrices

Here we review several notions and lemmas due to Golovnev, Regev, and Weinstein [15]. For a matrix M over a field $\mathbb{F}$, let $s(M)$ denote its sparsity, that is, the number of its nonzero entries. We say that a matrix M over $\mathbb{F}$ with rank k contains an ℓ-sparse column (row) basis if M contains k linearly independent columns (rows) with a total of at most ℓ nonzero entries. We first state a lemma that provides an upper bound on the number of matrices with sparse column and row bases.

▶ **Lemma 7** ([15]). *The number of rank k matrices in $\mathbb{F}^{n \times n}$ that contain ℓ-sparse column and row bases is at most $(n \cdot |\mathbb{F}|)^{6\ell}$.*

The following lemma relates the sparsity of a matrix with nonzero entries on the main diagonal to its rank.

▶ **Lemma 8** ([15]). *For every rank k matrix $M \in \mathbb{F}^{n \times n}$ with nonzero entries on the main diagonal,*

$$s(M) \geq \frac{n^2}{4k}.$$

We also need the following notion. An (n, k, s, ℓ)-*matrix* over a field $\mathbb{F}$ is a matrix in $\mathbb{F}^{n\times n}$ of rank k and sparsity s that contains ℓ-sparse column and row bases and has nonzero entries on the main diagonal. Note that by Lemma 8, an (n, k, s, ℓ)-matrix exists only if $s \geq \frac{n^2}{4k}$. For integers n, k, s' and a field $\mathbb{F}$ (which will always be clear from the context), let $\mathcal{M}_{n,k}^{(s')}$ be the collection that consists of all $(n', k', s', \frac{2s'k'}{n'})$-matrices over $\mathbb{F}$ for all $n' \in [n]$ and $k' \in [k]$ such that $\frac{k'}{n'} \leq \frac{k}{n}$. This collection is motivated by the following lemma.

▶ **Lemma 9** ([15]). *Every matrix in $\mathbb{F}^{n\times n}$ with rank at most k and nonzero entries on the main diagonal has a principal sub-matrix that lies in $\mathcal{M}_{n,k}^{(s')}$ for some s'.*

Now, for integers n, k, s', let $\mathcal{P}_{n,k}^{(s')}$ be the collection that consists of all pairs (M, R) such that, for some $n' \in [n]$, M is an $n' \times n'$ matrix in $\mathcal{M}_{n,k}^{(s')}$ and R is an n'-subset of $[n]$. Observe that Lemma 9 implies that for every digraph G on the vertex set $[n]$ with $\text{minrk}_{\mathbb{F}}(G) \leq k$ there exist s' and a pair (M, R) in $\mathcal{P}_{n,k}^{(s')}$ such that M represents the induced subgraph $G[R]$ of G on R, with respect to the natural order of the vertices in R (from smallest to largest).

The following lemma provides an upper bound on the size of $\mathcal{P}_{n,k}^{(s')}$.

▶ **Lemma 10.** *For every integers n, k, s', $|\mathcal{P}_{n,k}^{(s')}| \leq (n \cdot |\mathbb{F}|)^{24s'k/n}$.*

Proof. To bound the size of $\mathcal{P}_{n,k}^{(s')}$, we consider for every $n' \in [n]$ and $k' \in [k]$ such that $\frac{k'}{n'} \leq \frac{k}{n}$ the pairs (M, R) where M is an $(n', k', s', \frac{2s'k'}{n'})$-matrix and R is an n'-subset of $[n]$. By Lemma 7 there are at most $(n' \cdot |\mathbb{F}|)^{12s'k'/n'}$ such matrices M, each of which occurs in $\binom{n}{n'}$ pairs of $\mathcal{P}_{n,k}^{(s')}$. It follows that

$$\begin{aligned}
|\mathcal{P}_{n,k}^{(s')}| &\leq \sum_{n',k'} \binom{n}{n'} \cdot (n' \cdot |\mathbb{F}|)^{12s'k'/n'} \leq n^2 \cdot \max_{n',k'} \left(n^{n'} \cdot (n' \cdot |\mathbb{F}|)^{12s'k'/n'}\right) \\
&\leq \max_{n',k'} \left(n^{3n'} \cdot (n' \cdot |\mathbb{F}|)^{12s'k'/n'}\right) \leq \max_{n',k'} \left((n \cdot |\mathbb{F}|)^{3n'+12s'k'/n'}\right) \\
&\leq \max_{n',k'} \left((n \cdot |\mathbb{F}|)^{12s'k'/n'+12s'k'/n'}\right) \leq (n \cdot |\mathbb{F}|)^{24s'k/n},
\end{aligned}$$

where in the fifth inequality we have used the relation $s' \geq \frac{n'^2}{4k'}$ from Lemma 8, and in the sixth we have used $\frac{k'}{n'} \leq \frac{k}{n}$. ◀

3.3 Proof of Theorem 2

We prove the following theorem and then derive Theorem 2. Recall that for a graph H with $h \geq 3$ vertices and $f \geq 3$ edges, we denote $\gamma(H) = \frac{h-2}{f-1}$. We also let $\exp(x)$ stand for e^x.

▶ **Theorem 11.** *For every graph H with at least 3 edges there exists $c = c(H) > 0$ such that for every integer n and a finite field $\mathbb{F}$,*

$$g(n, H, \mathbb{F}) \geq c \cdot \frac{n^{1-\gamma(H)}}{\log(n \cdot |\mathbb{F}|)}.$$

Proof. Fix a graph H with $h \geq 3$ vertices and $f \geq 3$ edges and denote $\gamma = \gamma(H) = \frac{h-2}{f-1} > 0$. The proof is via the probabilistic method. Let $\vec{G} \sim \vec{G}(n, p)$ be a random digraph on the vertex set $[n]$ where each directed edge is taken randomly and independently with probability p. Set $q = 1 - p$. Let G be the (undirected) graph on $[n]$ in which two distinct vertices i, j are adjacent if both the directed edges (i, j) and (j, i) are included in $\vec{G}$. Notice that every

two distinct vertices are adjacent in G with probability p^2 independently of the adjacencies between other vertex pairs.

To prove the theorem, we will show that for a certain choice of p the random graph G satisfies with positive probability that its complement $\overline{G}$ is H-free and that $\text{minrk}_{\mathbb{F}}(G) > k$, where

$$k = c_1 \cdot \frac{n^{1-\gamma}}{\ln(n \cdot |\mathbb{F}|)} \tag{3}$$

for a constant $c_1 > 0$ that depends only on H. To do so, we define two families of events as follows.

First, for every set $I \subseteq [n]$ of size $|I| = h$, let A_I be the event that the induced subgraph of $\overline{G}$ on I contains a copy of H. Observe that

$$\Pr[A_I] \leq h! \cdot (1-p^2)^f = h! \cdot (1-(1-q)^2)^f \leq h! \cdot (2q)^f.$$

Second, consider the collection $\mathcal{P} = \cup_{s' \in [n^2]} \mathcal{P}_{n,k}^{(s')}$ (see Section 3.2). Recall that every element of $\mathcal{P}$ is a pair (M, R) such that, for some $n' \in [n]$, M is an $n' \times n'$ matrix over $\mathbb{F}$ and R is an n'-subset of $[n]$. Denote $N_{s'} = |\mathcal{P}_{n,k}^{(s')}|$. By Lemma 10, combined with (3), we have

$$N_{s'} \leq (n \cdot |\mathbb{F}|)^{24s'k/n} = \exp(24c_1 \cdot s' \cdot n^{-\gamma}). \tag{4}$$

Let $\mathcal{S} = \{s' \in [n^2] \mid N_{s'} \geq 1\}$. By Lemma 8, for every $s' \in \mathcal{S}$ and an $n' \times n'$ matrix of rank k' in $\mathcal{M}_{n,k}^{(s')}$ where $n' \in [n]$, $k' \in [k]$, and $\frac{k'}{n'} \leq \frac{k}{n}$, we have that

$$s' \geq \frac{n'}{4} \cdot \frac{n'}{k'} \geq \frac{n'}{4} \cdot \frac{n}{k} = n' \cdot \frac{n^{\gamma} \cdot \ln(n \cdot |\mathbb{F}|)}{4c_1}. \tag{5}$$

Now, for every pair $(M, R) \in \mathcal{P}$, let $B_{M,R}$ be the event that the matrix M represents over $\mathbb{F}$ the induced subgraph $\vec{G}[R]$ of $\vec{G}$ on R with respect to the natural order of the vertices in R. For M to represent $\vec{G}[R]$ we require that for every distinct i, j such that $M_{i,j} \neq 0$, there is an edge in $\vec{G}$ from the ith to the jth vertex of R. Hence, for $M \in \mathbb{F}^{n' \times n'}$ of sparsity s' and an n'-subset R of $[n]$,

$$\Pr[B_{M,R}] = p^{s'-n'} \leq p^{s'/2} = (1-q)^{s'/2} \leq \exp(-qs'/2),$$

where for the first inequality we have used the inequality $s' \geq 2n'$ which follows from (5) for every sufficiently large n.

We claim that it suffices to prove that with positive probability none of the events A_I and $B_{M,R}$ holds. Indeed, this implies that there exists an n-vertex digraph $\vec{G}$ that does not satisfy any of these events. Since the A_I's are not satisfied it immediately follows that the complement $\overline{G}$ of the (undirected) graph G associated with $\vec{G}$ is H-free. We further claim that $\text{minrk}_{\mathbb{F}}(G) > k$. To see this, assume by contradiction that there exists a matrix $M \in \mathbb{F}^{n \times n}$ of rank at most k that represents G, and thus, in particular, represents $\vec{G}$. By Lemma 9, such an M has a principal $n' \times n'$ sub-matrix $M' \in \mathcal{M}_{n,k}^{(s')}$ for some n' and s'. Hence, for some n'-subset R of $[n]$, the matrix M' represents $\vec{G}[R]$ with respect to the natural order of the vertices in R, in contradiction to the fact that the event $B_{M',R}$ with $(M', R) \in \mathcal{P}$ does not hold.

To prove that with positive probability none of the events A_I and $B_{M,R}$ holds, we apply the Lovász Local Lemma (Lemma 6). To this end, construct a (symmetric) dependency digraph $D = (V, E)$ whose vertices represent all the events A_I and $B_{M,R}$, and whose edges are defined as follows.

- An A_I-vertex and an $A_{I'}$-vertex are joined by edges (in both directions) if $|I \cap I'| \geq 2$. Notice that the events A_I and $A_{I'}$ are independent when $|I \cap I'| < 2$.
- An A_I-vertex and a $B_{M,R}$-vertex are joined by edges if there are distinct $i, j \in I \cap R$ for which the entry of M that corresponds to the edge (i, j) is nonzero. Notice that the events A_I and $B_{M,R}$ are independent when such i and j do not exist.
- Every two distinct $B_{M,R}$-vertices are joined by edges.

Clearly, each event is mutually independent of all other events besides those adjacent to it in D, and thus D is a dependency digraph for our events. Observe that every A_I-vertex is adjacent to at most $\binom{h}{2} \cdot \binom{n}{h-2} \leq \binom{h}{2} \cdot n^{h-2}$ $A_{I'}$-vertices. Additionally, every $B_{M,R}$-vertex, where M is an $n' \times n'$ matrix of sparsity s', is adjacent to at most $(s' - n') \cdot \binom{n}{h-2} < s' \cdot n^{h-2}$ A_I-vertices. Finally, every vertex of D is adjacent to at most $N_{s'}$ $B_{M,R}$-vertices with $M \in \mathcal{M}_{n,k}^{(s')}$ (that is, $s(M) = s'$).

To apply Lemma 6 we assign a number in $[0, 1)$ to each vertex of D. Define

$$q = c_2 \cdot n^{-\gamma}, \quad x = c_3 \cdot n^{-\gamma \cdot f}, \quad \text{and} \quad x_{s'} = \exp(-c_4 \cdot s' \cdot n^{-\gamma}) \quad \text{for every } s' \in \mathcal{S},$$

where $c_2, c_3, c_4 > 0$ are constants, depending only on H, to be determined. We assign the number x to every A_I-vertex, and the number $x_{s'}$ to every $B_{M,R}$-vertex with $s(M) = s'$. We present now the conditions of Lemma 6. For every A_I-vertex, recalling that $\Pr[A_I] \leq h! \cdot (2q)^f$, we require

$$h! \cdot (2q)^f \leq x \cdot (1 - x)^{\binom{h}{2} \cdot n^{h-2}} \cdot \prod_{s' \in \mathcal{S}} (1 - x_{s'})^{N_{s'}}. \tag{6}$$

Similarly, for every $B_{M,R}$-vertex with $s(M) = s'$, recalling that $\Pr[B_{M,R}] \leq \exp(-qs'/2)$, we require

$$\exp(-qs'/2) \leq x_{s'} \cdot (1 - x)^{s' \cdot n^{h-2}} \cdot \prod_{s' \in \mathcal{S}} (1 - x_{s'})^{N_{s'}}. \tag{7}$$

To complete the proof, it suffices to show that the constants $c_1, c_2, c_3, c_4 > 0$ can be chosen in a way that satisfies the inequalities (6) and (7). Consider the following three constraints:

1. $c_2 > 2 \cdot (2c_3 + c_4)$,
2. $c_3 \geq h! \cdot (2c_2)^f \cdot \exp(3)$, and
3. $c_4 \geq 32 \cdot c_1$.

It is easy to see that it is possible to choose the constants under the above constraints. Indeed, by $f \geq 3$, for a sufficiently small choice of $c_2 > 0$ one can take c_3 with, say, an equality in Item 2 so that some $c_4 > 0$ satisfies Item 1. Then, c_1 can be chosen as a positive constant satisfying Item 3. We show now that such a choice satisfies (6) and (7) for every sufficiently large n. Note that we use below several times the inequality $1 - \alpha \geq \exp(-2\alpha)$, which holds for any $\alpha \in [0, 1/2]$.

First, use (4) and the condition $c_4 \geq 32 \cdot c_1$ to obtain that

$$\begin{aligned} \sum_{s' \in \mathcal{S}} x_{s'} \cdot N_{s'} &\leq \sum_{s' \in \mathcal{S}} \exp((24c_1 - c_4) \cdot s' \cdot n^{-\gamma}) \\ &\leq \sum_{s' \in \mathcal{S}} \exp(-8c_1 \cdot s' \cdot n^{-\gamma}) \leq \sum_{s' \in \mathcal{S}} \exp(-2 \ln n) \leq 1, \end{aligned}$$

where the third inequality follows by $s' \geq \frac{n^\gamma \cdot \ln(n \cdot |\mathbb{F}|)}{4c_1}$ which we get from (5), and the fourth by $|\mathcal{S}| \leq n^2$. Considering the term $\prod_{s' \in \mathcal{S}} (1 - x_{s'})^{N_{s'}}$, which appears in both (6) and (7),

we derive that

$$\prod_{s' \in \mathcal{S}} (1 - x_{s'})^{N_{s'}} \geq \prod_{s' \in \mathcal{S}} \exp(-2x_{s'} \cdot N_{s'}) = \exp\Big(-2 \cdot \sum_{s' \in \mathcal{S}} x_{s'} \cdot N_{s'} \Big) \geq \exp(-2).$$

For inequality (6), observe that

$$\begin{aligned} x \cdot (1-x)^{\binom{h}{2} \cdot n^{h-2}} \cdot \prod_{s' \in \mathcal{S}} (1 - x_{s'})^{N_{s'}} &\geq x \cdot \exp\Big(-2x \cdot \binom{h}{2} \cdot n^{h-2} \Big) \cdot \exp(-2) \\ &= c_3 \cdot n^{-\gamma \cdot f} \cdot \exp\Big(-2c_3 \cdot n^{-\gamma \cdot f} \cdot \binom{h}{2} \cdot n^{h-2} - 2 \Big) \\ &\geq h! \cdot (2c_2)^f \cdot n^{-\gamma \cdot f} \cdot \exp\Big(1 - 2c_3 \cdot \binom{h}{2} \cdot n^{-\gamma} \Big) \\ &\geq h! \cdot (2q)^f, \end{aligned}$$

where for the second inequality we use $c_3 \geq h! \cdot (2c_2)^f \cdot \exp(3)$ and $\gamma = \frac{h-2}{f-1}$, and for the third we use the assumption that n is sufficiently large. For inequality (7), observe that

$$\begin{aligned} x_{s'} \cdot (1-x)^{s' \cdot n^{h-2}} \cdot \prod_{s' \in \mathcal{S}} (1 - x_{s'})^{N_{s'}} &\geq x_{s'} \cdot \exp(-2x \cdot s' \cdot n^{h-2}) \cdot \exp(-2) \\ &= \exp(-c_4 \cdot s' \cdot n^{-\gamma}) \cdot \exp(-2c_3 \cdot n^{-\gamma \cdot f} \cdot s' \cdot n^{h-2}) \cdot \exp(-2) \\ &= \exp(-(2c_3 + c_4) \cdot s' \cdot n^{-\gamma} - 2) \\ &\geq \exp(-(c_2/2) \cdot s' \cdot n^{-\gamma}) \\ &= \exp(-qs'/2), \end{aligned}$$

where for the second equality we again use the definition of γ, and for the second inequality we use the condition $c_2 > 2 \cdot (2c_3 + c_4)$, the fact that $s' \cdot n^{-\gamma} = \omega(1)$ by (5), and the assumption that n is sufficiently large. This completes the proof. ◀

We can derive now Theorem 2. Recall that $\gamma_0(H) = \min_{H'} \gamma(H')$, where the minimum is over all subgraphs H' of H with at least 3 edges.

Proof of Theorem 2. For a graph H with $h \geq 3$ vertices and $f \geq 3$ edges, let H' be a subgraph of H with at least 3 edges such that $\gamma_0(H) = \gamma(H')$. By Theorem 11 there exists $c > 0$ such that

$$g(n, H', \mathbb{F}) \geq c \cdot \frac{n^{1-\gamma_0(H)}}{\log(n \cdot |\mathbb{F}|)}$$

for every integer n and a finite field $\mathbb{F}$. Since every H'-free graph is also H-free, it follows that $g(n, H, \mathbb{F}) \geq g(n, H', \mathbb{F})$ and we are done. ◀

3.4 The Minrank of Graphs with Small Independence Number

For an integer $t \geq 3$, $g(n, K_t, \mathbb{F})$ is the maximum possible minrank over $\mathbb{F}$ of an n-vertex graph with independence number smaller than t. For this case we derive the following corollary.

▶ **Corollary 12.** *For every $t \geq 3$ there exists $c = c(t) > 0$ such that for every integer n and a finite field $\mathbb{F}$,*

$$g(n, K_t, \mathbb{F}) \geq c \cdot \frac{n^{1-\frac{2}{t+1}}}{\log(n \cdot |\mathbb{F}|)}.$$

Proof. Apply Theorem 2 to the graph $H = K_t$, and notice that $\gamma_0(K_t) = \gamma(K_t) = \frac{t-2}{\binom{t}{2}-1} = \frac{2}{t+1}$. ◀

For $H = K_3$, we observe that our lower bound on $g(n, K_3, \mathbb{F})$ is nearly tight.

▶ **Proposition 13.** *There exist constants $c_1, c_2 > 0$ such that for every integer n and a finite field $\mathbb{F}$,*

$$c_1 \cdot \frac{\sqrt{n}}{\log(n \cdot |\mathbb{F}|)} \leq g(n, K_3, \mathbb{F}) \leq c_2 \cdot \sqrt{\frac{n}{\log n}}.$$

Proof. For the lower bound apply Corollary 12 with $t = 3$. To prove the upper bound we need a result of Ajtai et al. [1] which says that every triangle-free n-vertex graph has an independent set of size $\Omega(\sqrt{n \cdot \log n})$. By repeatedly omitting such independent sets it follows that the chromatic number of such a graph is $O(\sqrt{n/\log n})$. Now, let G be an n-vertex graph whose complement $\overline{G}$ is triangle-free. We get that $\text{minrk}_{\mathbb{F}}(G) \leq \chi(\overline{G}) \leq O(\sqrt{n/\log n})$, as required. ◀

4 Non-bipartite Graphs

In this section we show that for every non-bipartite graph H there are H-free graphs with low minrank over $\mathbb{R}$, confirming Theorem 3. We start with the case where H is an odd cycle, and since every non-bipartite graph contains an odd cycle the general result follows easily. The proof is by an explicit construction from the following family of graphs.

▶ **Definition 14.** For integers $m \leq s \leq d$, the graph $K^<(d, s, m)$ is defined as follows: the vertices are all the s-subsets of $[d]$, and two distinct sets A, B are adjacent if $|A \cap B| < m$.

The minrank of such graphs over finite fields was recently studied in [18] using tools from [4]. The proof technique of [18] can be used for the real field as well, as shown below.

▶ **Proposition 15.** *For every integers $m \leq s \leq d$,*

$$\text{minrk}_{\mathbb{R}}(K^<(d, s, m)) \leq \sum_{i=0}^{s-m} \binom{d}{i}.$$

Proof. Let $f : \{0,1\}^d \times \{0,1\}^d \to \mathbb{R}$ be the function defined by

$$f(x, y) = \prod_{j=m}^{s-1} \left(\sum_{i=1}^{d} x_i y_i - j \right)$$

for every $x, y \in \{0,1\}^d$. Expanding f as a linear combination of monomials, the relation $z^2 = z$ for $z \in \{0,1\}$ implies that one can reduce to 1 the exponent of each variable occuring in a monomial. It follows that f can be represented as a multilinear polynomial in the $2d$ variables of x and y. By combining terms involving the same monomial in the variables of x, one can write f as

$$f(x, y) = \sum_{i=1}^{R} g_i(x) h_i(y)$$

for an integer R and functions $g_i, h_i : \{0,1\}^d \to \mathbb{R}$, $i \in [R]$, such that the g_i's are distinct multilinear monomials of total degree at most $s - m$ in d variables. It follows that $R \leq \sum_{i=0}^{s-m} \binom{d}{i}$.

Now, let M_1 and M_2 be the $2^d \times R$ matrices whose rows are indexed by $\{0,1\}^d$ and whose columns are indexed by $[R]$, defined by $(M_1)_{x,i} = g_i(x)$ and $(M_2)_{x,i} = h_i(x)$. Then, the matrix $M = M_1 \cdot M_2^T$ has rank at most R and for every $x, y \in \{0,1\}^d$ it holds that $M_{x,y} = f(x,y)$.

Finally, let V be the vertex set of $K^<(d,s,m)$, that is, the collection of all s-subsets of $[d]$, and identify every vertex $A \in V$ with an indicator vector $c_A \in \{0,1\}^d$ in the natural way. We claim that the matrix M restricted to $V \times V$ represents the graph $K^<(d,s,m)$. Indeed, for every $A, B \in V$ we have

$$M_{c_A,c_B} = f(c_A,c_B) = \prod_{j=m}^{s-1} \Big(|A \cap B| - j\Big).$$

Hence, for every $A \in V$ we have $|A| = s$ and thus $M_{c_A,c_A} \neq 0$, whereas for every distinct non-adjacent $A, B \in V$ we have $m \leq |A \cap B| \leq s-1$ and thus $M_{c_A,c_B} = 0$. Since the restriction of M to $V \times V$ has rank at most R it follows that $\mathrm{minrk}_{\mathbb{R}}(K^<(d,s,m)) \leq R$, and we are done. ◀

We turn to identify graphs $K^<(d,s,m)$ with no short odd cycles. For this purpose, take an even integer d, $s = \frac{d}{2}$, and $m = \varepsilon \cdot d$ for a small constant $\varepsilon > 0$. Every path in these graphs is a sequence of $\frac{d}{2}$-subsets of $[d]$ such that the intersection size of every two consecutive sets is small. This implies, for a sufficiently small ε, that the sets in the even positions of the path are almost disjoint from the first set, whereas the sets in the odd positions of the path share with it many elements, hence such a graph contains no short odd cycle. This is shown formally in the following lemma.

▶ **Lemma 16.** *Let $\ell \geq 3$ be an odd integer. For every even integer d and an integer $m \leq \frac{d}{2\ell}$, the graph $K^<(d, \frac{d}{2}, m)$ contains no odd cycle of length at most ℓ.*

Proof. Fix an odd integer $\ell \geq 3$, an even integer d, and an integer $m \leq \frac{d}{2\ell}$. We prove that for every odd integer ℓ', such that $3 \leq \ell' \leq \ell$, the graph $K^<(d, \frac{d}{2}, m)$ contains no cycle of length ℓ'. For such an ℓ', let $A_1, A_2, \ldots, A_{\ell'}$ be a sequence of ℓ' vertices in the graph, i.e., $\frac{d}{2}$-subsets of $[d]$. Assuming that for every $i \leq \ell' - 1$ the vertices A_i and A_{i+1} are adjacent in the graph, that is, $|A_i \cap A_{i+1}| < m$, our goal is to show that A_1 and $A_{\ell'}$ are not.

To this end, we argue that for every i, such that $0 \leq i \leq \frac{\ell'-1}{2}$, we have

$$|A_1 \cap A_{2i+1}| \geq \frac{d}{2} - 2i \cdot m. \tag{8}$$

We prove this claim by induction on i. The case $i = 0$ follows immediately from $|A_1| = \frac{d}{2}$. Assume that (8) holds for $i-1$, that is, $|A_1 \cap A_{2i-1}| \geq \frac{d}{2} - (2i-2) \cdot m$. Observe that this implies that

$$\begin{aligned} |A_1 \cap A_{2i}| &= |A_1 \cap A_{2i} \cap A_{2i-1}| + |A_1 \cap A_{2i} \cap \overline{A_{2i-1}}| \\ &\leq |A_{2i-1} \cap A_{2i}| + |A_1 \cap \overline{A_{2i-1}}| \\ &\leq m + |A_1| - |A_1 \cap A_{2i-1}| \\ &\leq m + \frac{d}{2} - \Big(\frac{d}{2} - (2i-2) \cdot m\Big) = (2i-1) \cdot m, \end{aligned}$$

where in the second inequality we have used $|A_{2i-1} \cap A_{2i}| < m$. We proceed by proving (8)

for i. Observe that

$$\begin{aligned} |A_1 \cap A_{2i+1}| &= |A_{2i+1}| - |\overline{A_1} \cap A_{2i+1}| \\ &= |A_{2i+1}| - |\overline{A_1} \cap A_{2i+1} \cap A_{2i}| - |\overline{A_1} \cap A_{2i+1} \cap \overline{A_{2i}}| \\ &\geq \frac{d}{2} - m - |\overline{A_1} \cap \overline{A_{2i}}|, \end{aligned}$$

where we have used $|A_{2i} \cap A_{2i+1}| < m$. Notice that

$$|\overline{A_1} \cap \overline{A_{2i}}| = d - |A_1 \cup A_{2i}| = d - (|A_1| + |A_{2i}| - |A_1 \cap A_{2i}|) = |A_1 \cap A_{2i}|.$$

It follows that

$$|A_1 \cap A_{2i+1}| \geq \frac{d}{2} - m - |A_1 \cap A_{2i}| \geq \frac{d}{2} - m - (2i-1) \cdot m = \frac{d}{2} - 2i \cdot m,$$

completing the proof of (8).

Finally, applying (8) to $i = \frac{\ell'-1}{2}$, using the assumption $m \leq \frac{d}{2\ell}$, we get that

$$|A_1 \cap A_{\ell'}| \geq \frac{d}{2} - (\ell'-1) \cdot m = \frac{d}{2} - \ell' \cdot m + m \geq \frac{d}{2} - \ell \cdot m + m \geq m,$$

hence A_1 and $A_{\ell'}$ are not adjacent in the graph $K^<(d, \frac{d}{2}, m)$. It thus follows that the graph contains no cycle of length ℓ', as desired. ◀

Equipped with Proposition 15 and Lemma 16, we obtain the following.

▶ **Theorem 17.** *For every odd integer $\ell \geq 3$ there exists $\delta = \delta(\ell) > 0$ such that for every sufficiently large integer n, there exists an n-vertex graph G with no odd cycle of length at most ℓ such that*

$$\mathrm{minrk}_{\mathbb{R}}(G) \leq n^{1-\delta}.$$

Proof. Fix an odd integer $\ell \geq 3$. For an integer d divisible by 2ℓ, consider the graph $G = K^<(d, \frac{d}{2}, m)$ where $m = \frac{d}{2\ell}$. By Lemma 16, G contains no odd cycle of length at most ℓ. As for the minrank, Proposition 15 implies that

$$\mathrm{minrk}_{\mathbb{R}}(G) \leq \sum_{i=0}^{d/2-m} \binom{d}{i} \leq 2^{H(\frac{1}{2} - \frac{m}{d}) \cdot d} = 2^{H(\frac{1}{2} - \frac{1}{2\ell}) \cdot d},$$

where H stands for the binary entropy function. Since G has $|V| = \binom{d}{d/2} = 2^{(1-o(1)) \cdot d}$ vertices, for any $\delta > 0$ such that $H(\frac{1}{2} - \frac{1}{2\ell}) < 1 - \delta$ we have $\mathrm{minrk}_{\mathbb{R}}(G) \leq |V|^{1-\delta}$ for every sufficiently large integer d. The proof is completed by considering, for every sufficiently large integer n, some n-vertex subgraph of the graph defined above, where d is the smallest integer divisible by 2ℓ such that $n \leq \binom{d}{d/2}$. ◀

Now, Theorem 3 follows easily from Theorem 17.

Proof of Theorem 3. Let H be a non-bipartite graph. Then, for some odd integer $\ell \geq 3$, the cycle C_ℓ is a subgraph of H. By Theorem 17, there exists $\delta > 0$ such that for every sufficiently large integer n, there exists an n-vertex C_ℓ-free graph G satisfying $\mathrm{minrk}_{\mathbb{R}}(G) \leq n^{1-\delta}$. Since every C_ℓ-free graph is also H-free, the result follows. ◀

▶ Remark. As mentioned in the introduction, Theorem 3 implies a lower bound on $g(n, H, \mathbb{R})$ for every non-bipartite graph H (see Corollary 4). We note that upper bounds on certain Ramsey numbers can be used to derive upper bounds on $g(n, H, \mathbb{F})$ for a general field $\mathbb{F}$. For example, it was shown in [12] that for every $\ell \geq 3$, every n-vertex C_ℓ-free graph has an independent set of size $\Omega(n^{1-1/k})$ for $k = \lceil \frac{\ell}{2} \rceil$ (see [9, 30] for slight improvements). By repeatedly omitting such independent sets it follows that the chromatic number of such a graph is $O(n^{1/k})$. This implies that every n-vertex graph G whose complement is C_ℓ-free satisfies $\mathrm{minrk}_{\mathbb{F}}(G) \leq \chi(\overline{G}) \leq O(n^{1/k})$, hence $g(n, C_\ell, \mathbb{F}) \leq O(n^{1/k})$.

References

1. Miklós Ajtai, János Komlós, and Endre Szemerédi. A note on Ramsey numbers. *J. Comb. Theory, Ser. A*, 29(3):354–360, 1980.
2. Noga Alon. Explicit Ramsey graphs and orthonormal labelings. *Electr. J. Comb.*, 1(R12), 1994.
3. Noga Alon. Graph powers. In B. Bollobás, editor, *Contemporary Combinatorics*, Bolyai Society Mathematical Studies, pages 11–28. Springer, 2002.
4. Noga Alon, László Babai, and H. Suzuki. Multilinear polynomials and Frankl–Ray-Chaudhuri–Wilson type intersection theorems. *J. Comb. Theory, Ser. A*, 58(2):165–180, 1991.
5. Noga Alon, Michael Krivelevich, and Benny Sudakov. Maxcut in H-free graphs. *Combinatorics, Probability and Computing*, 14(5-6):629—-647, 2005.
6. Noga Alon and Joel H. Spencer. *The Probabilistic Method*. Wiley Publishing, 4th edition, 2016.
7. Ziv Bar-Yossef, Yitzhak Birk, T. S. Jayram, and Tomer Kol. Index coding with side information. In *FOCS*, pages 197–206, 2006.
8. Anna Blasiak, Robert Kleinberg, and Eyal Lubetzky. Broadcasting with side information: Bounding and approximating the broadcast rate. *IEEE Trans. Information Theory*, 59(9):5811–5823, 2013.
9. Yair Caro, Yusheng Li, Cecil C. Rousseau, and Yuming Zhang. Asymptotic bounds for some bipartite graph: complete graph Ramsey numbers. *Discrete Mathematics*, 220(1-3):51–56, 2000.
10. Eden Chlamtáč and Ishay Haviv. Linear index coding via semidefinite programming. *Combinatorics, Probability & Computing*, 23(2):223–247, 2014. Preliminary version in SODA'12.
11. Bruno Codenotti, Pavel Pudlák, and Giovanni Resta. Some structural properties of low-rank matrices related to computational complexity. *Theor. Comput. Sci.*, 235(1):89–107, 2000. Preliminary version in ECCC'97.
12. Paul Erdös, Ralph J. Faudree, Cecil C. Rousseau, and Richard H. Schelp. On cycle-complete graph Ramsey numbers. *J. Graph Theory*, 2(1):53–64, 1978.
13. Paul Erdös and László Lovász. Problems and results on 3-chromatic hypergraphs and some related questions. In A. Hajnal, R. Rado, and V. T. Sós, editors, *Infinite and Finite Sets*, pages 609–627. North-Holland, Amsterdam, 1975.
14. Paul Erdös, Robert J. McEliece, and Herbert Taylor. Ramsey bounds for graph products. *Pacific J. Math.*, 37(1):45–46, 1971.
15. Alexander Golovnev, Oded Regev, and Omri Weinstein. The minrank of random graphs. In *Randomization and Approximation Techniques in Computer Science (RANDOM)*, pages 46:1–46:13, 2017.
16. Willem Haemers. On some problems of Lovász concerning the Shannon capacity of a graph. *IEEE Trans. Inform. Theory*, 25(2):231–232, 1979.

17 Willem Haemers. An upper bound for the Shannon capacity of a graph. In *Algebraic methods in graph theory, Vol. I, II (Szeged, 1978)*, volume 25 of *Colloq. Math. Soc. János Bolyai*, pages 267–272. North-Holland, Amsterdam, 1981.

18 Ishay Haviv. On minrank and the Lovász theta function. In *Approximation Algorithms for Combinatorial Optimization Problems (APPROX)*, 2018. To appear.

19 Ishay Haviv and Michael Langberg. H-wise independence. In *Innovations in Theoretical Computer Science (ITCS'13)*, pages 541–552, 2013.

20 B. S. Kashin and S. V. Konyagin. Systems of vectors in Hilbert space. In *Number theory, mathematical analysis, and their applications*, volume 157, pages 64–67. Trudy Mat. Inst. Steklov., 1981.

21 S. V. Konyagin. Systems of vectors in Euclidean space and an extremal problem for polynomials. *Mat. Zametki*, 29(1):63–74, 1981.

22 László Lovász. On the Shannon capacity of a graph. *IEEE Trans. Inform. Theory*, 25(1):1–7, 1979.

23 René Peeters. Orthogonal representations over finite fields and the chromatic number of graphs. *Combinatorica*, 16(3):417–431, 1996.

24 Pavel Pudlák. Cycles of nonzero elements in low rank matrices. *Combinatorica*, 22(2):321–334, 2002.

25 Pavel Pudlák, Vojtech Rödl, and Jirí Sgall. Boolean circuits, tensor ranks, and communication complexity. *SIAM J. Comput.*, 26(3):605–633, 1997.

26 Søren Riis. Information flows, graphs and their guessing numbers. *Electr. J. Comb.*, 14(1), 2007.

27 Moshe Rosenfeld. Almost orthogonal lines in E^d. *DIMACS Series in Discrete Math.*, 4:489–492, 1991.

28 Claude E. Shannon. The zero error capacity of a noisy channel. *Institute of Radio Engineers, Transactions on Information Theory*, IT-2:8–19, 1956.

29 Joel Spencer. Asymptotic lower bounds for Ramsey functions. *Discrete Mathematics*, 20:69–76, 1977.

30 Benny Sudakov. A note on odd cycle-complete graph Ramsey numbers. *Electr. J. Comb.*, 9(1), 2002.

31 Leslie G. Valiant. Graph-theoretic arguments in low-level complexity. In *Mathematical Foundations of Computer Science (MFCS), 6th Symposium*, pages 162–176, 1977.

32 Leslie G. Valiant. Why is Boolean complexity theory difficult? In *Poceedings of the London Mathematical Society symposium on Boolean function complexity*, volume 169, pages 84–94, 1992.

33 Xiaodong Xu, Xie Zheng, Geoffrey Exoo, and Stanislaw P. Radziszowski. Constructive lower bounds on classical multicolor Ramsey numbers. *Electr. J. Comb.*, 11(1), 2004.

Preserving Randomness for Adaptive Algorithms

William M. Hoza[1]
Department of Computer Science, University of Texas at Austin, Austin, TX, USA
whoza@utexas.edu
https://orcid.org/0000-0001-5162-9181

Adam R. Klivans
Department of Computer Science, University of Texas at Austin, Austin, TX, USA
klivans@cs.utexas.edu

Abstract

Suppose Est is a randomized estimation algorithm that uses n random bits and outputs values in $\mathbb{R}^d$. We show how to execute Est on k adaptively chosen inputs using only $n + O(k \log(d+1))$ random bits instead of the trivial nk (at the cost of mild increases in the error and failure probability). Our algorithm combines a variant of the INW pseudorandom generator [12] with a new scheme for shifting and rounding the outputs of Est. We prove that modifying the outputs of Est is necessary in this setting, and furthermore, our algorithm's randomness complexity is near-optimal in the case $d \leq O(1)$. As an application, we give a randomness-efficient version of the Goldreich-Levin algorithm; our algorithm finds all Fourier coefficients with absolute value at least θ of a function $F : \{0,1\}^n \to \{-1,1\}$ using $O(n \log n) \cdot \text{poly}(1/\theta)$ queries to F and $O(n)$ random bits (independent of θ), improving previous work by Bshouty et al. [3].

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization

Keywords and phrases pseudorandomness, adaptivity, estimation

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.43

Related Version [10], https://arxiv.org/abs/1611.00783

Acknowledgements We thank David Zuckerman for observations about block decision trees. We thank an anonymous reviewer for pointing out Impagliazzo and Zuckerman's previous work on this subject [13, 11].

1 Introduction

Let Est be a randomized algorithm that estimates some quantity $\mu(C) \in \mathbb{R}^d$ when given input C. The canonical example is the case when C is a Boolean circuit, $d = 1$, $\mu(C) \stackrel{\text{def}}{=} \Pr_x[C(x) = 1]$, and Est estimates $\mu(C)$ by evaluating C at several randomly chosen points. Suppose that Est uses n random bits, and $\Pr[\|\mathsf{Est}(C) - \mu(C)\|_\infty > \varepsilon] \leq \delta$.

Furthermore, suppose we want to use Est as a *subroutine*, executing it on inputs $C_1, C_2, \ldots, C_k$, where each C_i is chosen adaptively based on the previous outputs of Est. The naïve implementation uses nk random bits and fails with probability at most $k\delta$.

In this work, we show how to generically improve the randomness complexity of any algorithm with this structure, without increasing the number of executions of Est, at the

[1] Supported by the NSF GRFP under Grant DGE-1610403 and by a Harrington Fellowship from the University of Texas at Austin.

expense of mild increases in the error and failure probability. Our algorithm efficiently finds $Y_1, \ldots, Y_k \in \mathbb{R}^d$ with $\|Y_i - \mu(C_i)\|_\infty \leq O(\varepsilon d)$ for every i, our algorithm has failure probability $k\delta + \gamma$ for any $\gamma > 0$, and our algorithm uses a total of $n + O(k \log(d+1) + (\log k)\log(1/\gamma))$ random bits.

1.1 The randomness steward model

We model the situation described above by imagining two interacting agents: the *owner* (who plays the role of the outer algorithm) chooses the inputs $C_1, \ldots, C_k$, while the *steward* (who replaces the direct execution of Est) provides the output vectors $Y_1, \ldots, Y_k \in \mathbb{R}^d$. The reader might find it useful to think of the steward as a trusted "cloud computing" service. To justify the names, imagine that the owner gives the steward "stewardship" over her random bits. The steward's job is to "spend" as little randomness as possible without sacrificing too much accuracy.

To describe the model more rigorously, say that a function $f : \{0,1\}^n \to \mathbb{R}^d$ is (ε, δ)-*concentrated at* $\mu \in \mathbb{R}^d$ if $\Pr_{X \in \{0,1\}^n}[\|f(X) - \mu\|_\infty > \varepsilon] \leq \delta$. In each round i, the chosen input C_i defines a concentrated function $f_i(X) \stackrel{\text{def}}{=} \mathsf{Est}(C_i, X)$, so it is equivalent to imagine that the owner picks an arbitrary concentrated function. In the following definition, ε' is the error of the steward, and δ' is its failure probability.

▶ **Definition 1.** An (ε', δ')-*steward for* k *adaptively chosen* (ε, δ)-*concentrated functions* $f_1, \ldots, f_k : \{0,1\}^n \to \mathbb{R}^d$ is a randomized algorithm S that interacts with an *owner* O according to the following protocol.

1. For $i = 1$ to k:
   a. O chooses $f_i : \{0,1\}^n \to \mathbb{R}^d$ that is (ε, δ)-concentrated at some point $\mu_i \in \mathbb{R}^d$ and gives it to S.
   b. S chooses $Y_i \in \mathbb{R}^d$ and gives it to O.

Write $\mathsf{O} \leftrightarrow \mathsf{S}$ ("the interaction of O with S") to denote the above interaction. The requirement on S is that for all O,

$$\Pr[\max_i \|Y_i - \mu_i\|_\infty > \varepsilon' \text{ in } \mathsf{O} \leftrightarrow \mathsf{S}] \leq \delta'. \tag{1}$$

The probability is taken over the internal randomness of S and O.

From an information-theoretic perspective, stewards as defined above are not particularly interesting, because S could exhaustively examine all outputs of f_i to deterministically compute a point Y_i where f_i is concentrated. But we would like to avoid executing Est more than k times in total, so we will restrict attention to *one-query stewards*:

▶ **Definition 2.** A *one-query steward* is a steward that only accesses each f_i by querying it at a single point $X_i \in \{0,1\}^n$. (The point X_i is not seen by the owner.)

1.2 Our results

1.2.1 Main result: A one-query steward with good parameters

Our main result is the explicit construction of a one-query steward that simultaneously achieves low error, low failure probability, and low randomness complexity:

ε'	δ'	Randomness complexity	Reference
ε	$k\delta$	nk	Naïve
$O(\varepsilon d)$	$k\delta+\gamma$	$n+O(k\log(d+1)+(\log k)\log(1/\gamma))$	Theorem 3 (main)
$O(\varepsilon)$	$k\delta+\gamma$	$n+O(kd+(\log k)\log(1/\gamma))$	Full version [10]
$O(\varepsilon d)$	$k\delta+\gamma$	$n+k\log(d+2)+2\log(1/\gamma)+O(1)$	Full version [10][a]
$O(\varepsilon d)$	$2^{O(k\log(d+1))}\cdot\delta$	n	Full version [10]
$O(\varepsilon)$	$2^{O(kd)}\cdot\delta$	n	Full version [10]
$O(\varepsilon kd/\gamma)$	$k\delta+\gamma$	$n+O(k\log k+k\log d+k\log(1/\gamma))$	Based on [21]
$O(\varepsilon)$	$k\delta+k\cdot 2^{-n^{\Omega(1)}}$	$O(n^6+kd)$	Based on [13]
Any	Any ≤ 0.2	$n+\Omega(k)-\log(\delta'/\delta)$ (lower bound)	Full version [10]

[a] Computationally inefficient.

Figure 1 Upper and lower bounds for one-query stewards. Recall that ε,δ are the concentration parameters of $f_1,\dots,f_k$ (i.e. the error and failure probability of the estimation algorithm Est); ε',δ' are the error and failure probability of the steward S; n is the number of input bits to each f_i (i.e. the number of random coins used by Est); k is the number of rounds of adaptivity; d is the dimension of the output of each f_i (i.e. the dimension of the output of Est). Everywhere it appears, γ denotes an arbitrary positive number.

▶ **Theorem 3.** *For any $n,k,d\in\mathbb{N}$ and any $\varepsilon,\delta,\gamma>0$, there exists a one-query $(O(\varepsilon d),k\delta+\gamma)$-steward for k adaptively chosen (ε,δ)-concentrated functions $f_1,\dots,f_k:\{0,1\}^n\to\mathbb{R}^d$ with randomness complexity*

$$n+O(k\log(d+1)+(\log k)\log(1/\gamma)). \tag{2}$$

The total running time of the steward is $\mathrm{poly}(n,k,d,\log(1/\varepsilon),\log(1/\gamma))$.

We also give several variant stewards that achieve tradeoffs in parameters. (See Figure 1.)

1.2.2 Application: Acceptance probabilities of Boolean circuits

A natural application of Theorem 3 is a time- and randomness-efficient algorithm for estimating the acceptance probabilities of many adaptively chosen Boolean circuits.

▶ **Corollary 4.** *There exists a randomized algorithm with the following properties. Initially, the algorithm is given parameters $n,k\in\mathbb{N}$ and $\varepsilon,\delta>0$. Then, in round i $(1\leq i\leq k)$, the algorithm is given a Boolean circuit C_i on n input bits and outputs a number $Y_i\in[0,1]$. Here, C_i may be chosen adversarially based on $Y_1,\dots,Y_{i-1}$. With probability $1-\delta$, every Y_i is $\mu(C_i)\pm\varepsilon$, where $\mu(C_i)\stackrel{\mathrm{def}}{=}\Pr_x[C_i(x)=1]$. The total running time of the algorithm is*

$$O\left(\frac{\log k+\log(1/\delta)}{\varepsilon^2}\cdot\sum_{i=1}^{k}\mathrm{size}(C_i)\right)+\mathrm{poly}(n,k,1/\varepsilon,\log(1/\delta)), \tag{3}$$

and the total number of random bits used by the algorithm is $n+O(k+(\log k)\cdot\log(1/\delta))$.

Corollary 4 should be compared to the case when $C_1,\dots,C_k$ are chosen nonadaptively, for which the randomness complexity can be improved to $n+O(\log k+\log(1/\delta))$ by applying the Goldreich-Wigderson randomness-efficient sampler for Boolean functions [9] and reusing randomness. The proof of Corollary 4 works by combining the GW sampler with our steward. We also give similar algorithms for simulating an oracle for **promise-BPP** or **APP**.

1.2.3 Application: The Goldreich-Levin algorithm

For our main application, we give a randomness-efficient version of the Goldreich-Levin algorithm [8] (otherwise known as the Kushilevitz-Mansour algorithm [14]) for finding noticeably large Fourier coefficients. Given oracle access to $F : \{0,1\}^n \to \{-1,1\}$, for any $\theta > 0$, we show how to efficiently find a list containing all U with $|\widehat{F}(U)| \geq \theta$. (Alternatively, thinking of F as an exponentially long bitstring $F \in \{-1,1\}^{2^n}$, our algorithm finds all Hadamard codewords that agree with F in a $\left(\frac{1}{2}+\theta\right)$-fraction of positions.) Our algorithm makes $O(n\log(n/\delta)) \cdot \mathrm{poly}(1/\theta)$ queries to F, uses $O(n + (\log n)\log(1/\delta))$ random bits, and has failure probability δ. The number of random bits *does not depend on* θ. To achieve such a low randomness complexity, we first improve the randomness efficiency of each estimate in the Goldreich-Levin algorithm using the GW sampler. Then, we reduce the number of rounds of adaptivity by a factor of $\log(1/\theta)$ by making many estimates within each round. Interestingly, we apply our steward with $d = \mathrm{poly}(1/\theta)$, unlike the proof of Corollary 4 where we choose $d = 1$. (Recall that d is the number of real values estimated in each round.)

1.2.4 Randomness complexity lower bound

We prove a randomness complexity lower bound of $n + \Omega(k) - \log(\delta'/\delta)$ for any one-query steward. In the case $d \leq O(1)$, this comes close to matching our upper bounds. For example, to achieve $\delta' \leq O(k\delta)$, this lower bound says that $n + \Omega(k)$ random bits are needed; our main steward (Theorem 3) achieves $\varepsilon' \leq O(\varepsilon), \delta' \leq O(k\delta)$ using $n + O(k + (\log k)\log(1/\delta))$ random bits. At the other extreme, if we want a one-query steward that uses only n random bits, this lower bound says that the failure probability will be $\delta' \geq \exp(\Omega(k)) \cdot \delta$; one of our variant stewards uses n random bits to achieve $\varepsilon' \leq O(\varepsilon)$ and $\delta' \leq \exp(O(k)) \cdot \delta$.

1.3 Techniques

1.3.1 Block decision trees

A key component in the proof of our main result (Theorem 3) is a pseudorandom generator (PRG) for a new model that we call the *block decision tree* model. Informally, a block decision tree is a decision tree that reads its input from left to right, n bits at a time:

▶ **Definition 5.** For a finite alphabet Σ, a (k, n, Σ) *block decision tree* is a rooted tree $T = (V, E)$ of height k in which every node v at depth $< k$ has exactly $|\Sigma|$ children (labeled with the symbols in Σ) and has an associated function $v : \{0,1\}^n \to \Sigma$. We identify T with a function $T : (\{0,1\}^n)^{\leq k} \to V$ defined recursively: $T(\text{the empty string}) = \text{the root node}$, and if $T(X_1, \ldots, X_{i-1}) = v$, then $T(X_1, \ldots, X_i)$ is the child of v labeled $v(X_i)$.

The standard nonconstructive argument shows that there exists a γ-PRG for block decision trees with seed length $n + k\log|\Sigma| + 2\log(1/\gamma) + O(1)$. (See Appendix A.1 for the definition of a PRG in this setting.) We construct an explicit PRG with a nearly matching seed length:

▶ **Theorem 6.** *For every $n, k \in \mathbb{N}$, every finite alphabet Σ, and every $\gamma > 0$, there exists a γ-PRG $\mathsf{Gen} : \{0,1\}^s \to \{0,1\}^{nk}$ for (k, n, Σ) block decision trees with seed length*

$$s \leq n + O(k\log|\Sigma| + (\log k)\log(1/\gamma)). \tag{4}$$

The PRG can be computed in $\mathrm{poly}(n, k, \log|\Sigma|, \log(1/\gamma))$ time.

We prove Theorem 6 by modifying the INW generator for space-bounded computation [12].

1.3.2 Shifting and rounding

1.3.2.1 PRGs alone are not enough

Interestingly, to design good stewards, the standard technique of replacing truly random bits with pseudorandom bits is not sufficient. That is, define a *pseudorandom generation steward* to be a steward that simply queries each f_i at a pseudorandomly chosen point X_i and returns $Y_i = f_i(X_i)$. In the full version of this paper [10], we show that any pseudorandom generation steward must use at least $\Omega(nk)$ random bits, assuming $\delta' \leq 1/2$ and $\delta \geq 2^{-n/2+1}$.

1.3.2.2 Deliberately introducing error

To circumvent this $\Omega(nk)$ lower bound, our steward is forced to *modify the outputs* of $f_1, \ldots, f_k$. In each round, our main steward queries f_i at a pseudorandom point X_i, chooses a nearby value $Y_i \approx f_i(X_i)$, and returns this modified value Y_i to the owner O. The motivation for this idea is that by deliberately introducing a small amount of error, we reduce the amount of information about X_i that is leaked by Y_i. This way, we can recycle some of the randomness of X_i for future rounds.

To be more specific, for a steward S, let $\mathsf{S}(X)$ denote S using randomness X. Our main steward is of the form $\mathsf{S}(X) \stackrel{\text{def}}{=} \mathsf{S}_0(\mathsf{Gen}(X))$. Here, Gen is our PRG for block decision trees, and S_0 is a randomness-inefficient one-query steward. In each round, S_0 queries f_i at a fresh random point $X_i \in \{0,1\}^n$, but S_0 computes the return value Y_i by carefully *shifting and rounding* each coordinate of $f_i(X_i)$. This deterministic shifting and rounding procedure and its analysis are our main technical contributions.

1.3.2.3 Relation to block decision trees

To explain why this works, observe that when any steward and owner interact, it is natural to model the owner's behavior by a decision tree that branches at each node based on the value Y_i provided by the steward. The *branching factor* of this decision tree is a simple measure of the amount of information leaked. If S_0 simply returned $f_i(X_i)$ without any shifting or rounding, the branching factor for $\mathsf{O} \leftrightarrow \mathsf{S}_0$ would be 2^n. Three ideas dramatically reduce this branching factor.

- The first idea is to round. Suppose that S_0 rounded each coordinate of $f_i(X_i)$ to the nearest multiple of 2ε (with no shifting). Then the branching factor would be reduced to $2^d + \delta 2^n$. The $\delta 2^n$ term corresponds to the outputs of f_i that are far from its concentration point μ_i. The 2^d term corresponds to outputs $f_i(X_i)$ that are close to μ_i; if each coordinate of μ_i is approximately equidistant from two multiples of 2ε, then the corresponding coordinate of $f_i(X_i)$ could be rounded to either of those two values.
- The second idea is to *shift* each coordinate of $f_i(X_i)$ before rounding. In particular, S_0 finds a single value Δ_i such that after adding $\Delta_i \cdot 2\varepsilon$ to each coordinate of $f_i(X_i)$, every coordinate is ε-far from every rounding boundary. Then, S_0 rounds the shifted coordinates to obtain Y_i. This procedure reduces the branching factor down to $d+1+\delta 2^n$. To understand why, think of Δ_i as a *compressed* representation of Y_i. Assuming $f_i(X_i)$ is close to μ_i, given unlimited computation time, O could recover Y_i from Δ_i by computing the *true* vector μ_i, shifting *it* according to Δ_i, and rounding. Hence, each node of the tree just needs to have one child for each possible Δ_i value (along with the $\delta 2^n$ children for the case that $f_i(X_i)$ is far from μ_i).
- The third idea is to *relax* the requirement that the tree perfectly computes $\mathsf{O} \leftrightarrow \mathsf{S}_0$. In particular, for every owner O, we construct a block decision tree T_{O} that merely *certifies*

correctness of $\mathsf{O} \leftrightarrow \mathsf{S}_0$. That is, for any $X_1, \ldots, X_k$, if the node $T_{\mathsf{O}}(X_1, \ldots, X_k)$ indicates "success", then the error $\max_i \|Y_i - \mu_i\|_\infty$ in $\mathsf{O} \leftrightarrow \mathsf{S}_0(X_1, \ldots, X_k)$ is small. On the other hand, if $T_{\mathsf{O}}(X_1, \ldots, X_k)$ does not indicate success, then "all bets are off": the error $\max_i \|Y_i - \mu_i\|_\infty$ in $\mathsf{O} \leftrightarrow \mathsf{S}_0(X_1, \ldots, X_k)$ may be small or large. Our certification tree has the additional property that

$$\Pr_{X_1,\ldots,X_k}[T_{\mathsf{O}}(X_1, \ldots, X_k) \text{ indicates success}] \geq 1 - k\delta. \tag{5}$$

This relaxation allows us to reduce the branching factor down to just $d + 2$, because for each node, the $\delta 2^n$ children corresponding to outputs of f_i that are far from μ_i can all be merged into a single "failure" node.

Putting everything together, to save random bits, we don't need to try to fool $\mathsf{O} \leftrightarrow \mathsf{S}_0$. Instead, it suffices for Gen to fool the certification tree T_{O}. The small branching factor of T_{O} allows Gen to have a correspondingly small seed length.

1.4 Why can't we just reuse the random bits?

Notwithstanding our lower bounds, the reader might be tempted to think that randomness stewards are trivial: why not just pick $X \in \{0,1\}^n$ uniformly at random *once* and reuse it in every round? For the purpose of discussion, let us generalize, and suppose we are trying to execute an n-coin algorithm A (not necessarily an estimation algorithm) on k inputs $C_1, \ldots, C_k$. If $C_1, \ldots, C_k$ are chosen *non-adaptively* (i.e. all in advance), then we really can use the same X for each execution. By the union bound, the probability that $\mathsf{A}(C_i, X)$ fails for any i is at most $k\delta$.

That argument breaks down in the adaptive case, because C_2 is chosen based on $\mathsf{A}(C_1, X)$, and hence C_2 may be *stochastically dependent* on X, so $\mathsf{A}(C_2, X)$ is not guaranteed to have a low failure probability. For example, if X is encoded in the output $\mathsf{A}(C_1, X)$, then an adversarially chosen C_2 could *guarantee* that $\mathsf{A}(C_2, X)$ fails.

Even if $C_1, \ldots, C_k$ are chosen adaptively, randomness can be safely reused in an important special case: Suppose A is a **BPP** algorithm. Then we can let $\widehat{C}_1, \widehat{C}_2, \ldots, \widehat{C}_k$ be the inputs that would be chosen if A never failed. Then each $\widehat{C}_i$ really is independent of X, so by the union bound, with probability $1 - k\delta$, $\mathsf{A}(\widehat{C}_i, X)$ does not fail for any i. But if $\mathsf{A}(\widehat{C}_i, X)$ does not fail for any i, then by induction, $C_i = \widehat{C}_i$ for every i. So the overall failure probability is once again at most $k\delta$.

More generally, randomness can be safely reused if A is *pseudodeterministic*, i.e. for each input, there is a unique correct output that A gives with probability $1 - \delta$.[2] (Pseudodeterministic algorithms were introduced by Gat and Goldwasser [7].) A **BPP** algorithm is trivially pseudodeterministic.

In the standard Goldreich-Levin algorithm, randomness is used to estimate $\sum_{U \in \mathcal{U}} \widehat{F}(U)^2$ for certain collections of subsets $\mathcal{U}$. The algorithm's behavior depends on how the estimate compares to $\theta^2/2$. This process is not pseudodeterministic, because if the true value $\sum_{U \in \mathcal{U}} \widehat{F}(U)^2$ is very close to $\theta^2/2$, the estimate falls on each side of $\theta^2/2$ with noticeable probability.

[2] These two conditions (inputs are chosen non-adaptively, A is pseudodeterministic) are both special cases of the following condition under which the randomness X may be safely reused: for every $1 \leq i \leq k$, C_i is a pseudodeterministic function of $(C_0, C_1, \ldots, C_{i-1})$, where C_0 is a random variable that is independent of X.

1.5 Related work

1.5.1 Adaptive data analysis

The notion of a randomness steward is inspired by the closely related *adaptive data analysis* problem, introduced by Dwork et al. [6]. In the simplest version of this problem, there is an unknown distribution $\mathcal{D}$ over $\{0,1\}^n$ and a *data analyst* who wishes to estimate the mean values (with respect to $\mathcal{D}$) of k adaptively chosen functions $f_1, \ldots, f_k : \{0,1\}^n \to [0,1]$ using as few samples from $\mathcal{D}$ as possible. In this setting, these samples are held by a *mechanism* and are not directly accessible by the data analyst. In round i, the data analyst gives f_i to the mechanism, and the mechanism responds with an estimate of $\mathrm{E}_{x\sim\mathcal{D}}[f_i(x)]$. The mechanism constructs the estimate so as to leak as little information as possible about the sample, so that the same sample points can be safely reused for future estimates.

The data analyst and mechanism in the adaptive data analysis setting are analogous to the owner O and steward S in our setting, respectively. In each case, the idea is that the mechanism or steward can intentionally introduce a small amount of error into each estimate to hide information and thereby facilitate future estimates. Note, however, that in the adaptive data analysis problem, there is just one unknown distribution $\mathcal{D}$ and we are concerned with sample complexity, whereas in the randomness stewardship problem, we can think of each concentrated function f_i as defining a new distribution over $\mathbb{R}^d$ and we are concerned with randomness complexity.

1.5.2 The Saks-Zhou algorithm

Another highly relevant construction is the algorithm of Saks and Zhou [21] for simulating randomized log-space algorithms in deterministic space $O(\log^{3/2} n)$. The key component in this algorithm can be reinterpreted as a one-query randomness steward. This "Saks-Zhou steward" works by *randomly perturbing and rounding* the output of each f_i, and then reusing the same random query point X in each round. The perturbation and rounding are somewhat similar to our construction, but note that we shift the outputs of each f_i deterministically, whereas the Saks-Zhou steward uses random perturbations. The analysis of the Saks-Zhou steward is substantially different than the analysis of our steward. (See the full version of this paper [10] for the description and analysis of the Saks-Zhou steward.)

Our steward achieves better parameters than the Saks-Zhou steward (see Figure 1). In particular, to achieve failure probability $k\delta + \gamma$, the error ε' of the Saks-Zhou steward is $O(\varepsilon k d/\gamma)$ – the error grows linearly with k, the number of rounds of adaptivity, as well as with $1/\gamma$. This implies, for example, that if we tried to use the Saks-Zhou steward to estimate the acceptance probabilities of k adaptively chosen Boolean circuits to within $\pm\varepsilon$ with failure probability δ in a randomness-efficient way, we would need to evaluate each circuit on $\Theta(k^2\delta^{-2}\varepsilon^{-2}\log(k/\delta))$ inputs. In contrast, because of our steward's low error, the algorithm of Corollary 4 evaluates each circuit on just $O(\varepsilon^{-2}\log(k/\delta))$ inputs – an exponential improvement in both k and $1/\delta$. Furthermore, our steward has better randomness complexity than the Saks-Zhou steward.

1.5.3 Pseudorandom generators for adaptive algorithms

Impagliazzo and Zuckerman [13, 11] were the first to consider the problem of saving random bits when executing a randomized algorithm A on many adaptively chosen inputs. Instead of assuming that A is an estimation algorithm, Impagliazzo and Zuckerman's result assumes a known bound on the Shannon entropy of the output distribution of A (e.g., the number of bits output by A). They constructed a pseudorandom generator for this setting; for $k \gg n^6$, the seed length is approximately the sum of the entropy bounds for all the executions of A.

In contrast, we make no assumptions about the entropy of $\mathsf{Est}(C)$. Since $\mathsf{Est}(C)$ is a vector of arbitrary-precision real numbers, the entropy could be as large as n, the number of random bits used by Est. And indeed, the lower bound described in Section 1.3.2.1 implies that the approach of Impagliazzo and Zuckerman fails in our setting.

One might protest that the entropy of $\mathsf{Est}(C)$ can be reduced by simple rounding. In the full version of this paper [10], we construct and analyze a steward that straightforwardly rounds each output and then uses the Impagliazzo-Zuckerman generator in a black-box way. Our main steward achieves much better randomness complexity and failure probability than this "Impagliazzo-Zuckerman steward" (see Figure 1). The improvements come from our more powerful PRG and the fact that we shift before rounding.

1.5.4 Decision trees and branching programs

In the most common decision tree model, the branching factor $|\Sigma|$ is just 2, and each node reads an arbitrary bit of the input. In the more general *parity decision tree* model, each node computes the parity of some subset of the input bits. Kushilevitz and Mansour showed [14] that the Fourier ℓ_1 norm of any Boolean function computed by a parity decision tree is at most 2^k, the number of leaves in the tree. It follows immediately that any small-bias generator [17] fools parity decision trees with asymptotically optimal seed length.

Decision trees in which each node computes a more complicated function have also been studied previously. Bellare [2] introduced the *universal decision tree* model, in which each node computes an arbitrary Boolean function of the input bits. He gave a bound on the ℓ_1 norm of any Boolean function computed by a universal decision tree in terms of the ℓ_1 norms of the functions at each node. Unfortunately, his bound does not immediately imply any nontrivial PRGs for block decision trees.

A block decision tree can be thought of as a kind of space-bounded computation. Indeed, a block decision tree is a specific kind of *ordered branching program* of width $|\Sigma|^k$ and length k that reads n bits at a time. Hence, we could directly apply a PRG for ordered branching programs, such as the INW generator [12]. For these parameters, the INW generator has seed length of $n + O(k \log k \log |\Sigma| + \log k \log(1/\gamma))$. This seed length can be slightly improved by instead using Armoni's generator [1], but even that slightly improved seed length is larger than the seed length of the generator we construct.

1.5.5 Finding noticeably large Fourier coefficients

Our randomness-efficient version of the Goldreich-Levin algorithm should be compared to the results of Bshouty et al. [3], who gave several algorithms for finding noticeably large Fourier coefficients, all closely related to one another and based on an algorithm of Levin [15].

- Bshouty et al. gave one algorithm [3, Figure 4] that makes $O(\frac{n}{\theta^2} \log(\frac{n}{\delta\theta}))$ queries and uses $O(n \log(\frac{n}{\theta}) \log(\frac{1}{\delta\theta}))$ random bits. Our algorithm has better randomness complexity, but worse query complexity.
- Bshouty et al. gave another algorithm [3, Figure 5] that makes only $O(n/\theta^2)$ queries and uses just $O(\log(n/\theta) \cdot \log(1/\theta))$ random bits, but it merely outputs a list such that with probability $1/2$, some U in the list satisfies $|\widehat{F}(U)| \geq \theta$, assuming such a U exists.

We also remark that there is a *deterministic* version of the Goldreich-Levin algorithm for functions with bounded ℓ_1 norm; this follows easily from the work of Kushilevitz and Mansour [14] (see also [19, Section 6.4]). In contrast, our algorithm works for all functions $F : \{0,1\}^n \to \{-1,1\}$.

1. For $i = 1$ to k:
   a. O chooses $f_i : \{0,1\}^n \to \mathbb{R}^d$ and gives it to S_0.
   b. S_0 picks *fresh randomness* $X_i \in \{0,1\}^n$ and queries to obtain $W_i \stackrel{\text{def}}{=} f_i(X_i)$.
   c. S_0 computes Y_i by shifting and rounding W_i according to the algorithm in Section 2.1.
   d. S_0 gives Y_i to O.

Figure 2 Outline of $\mathsf{O} \leftrightarrow \mathsf{S}_0$.

1.6 Outline of this paper

In Section 2, we describe the shifting and rounding steward S_0 and prove that it admits certification trees with a small branching factor. In Section 3, we explain how to combine S_0 with our PRG to prove our main result (Theorem 3). In Section 4, we explain our randomness-efficient Goldreich-Levin algorithm. In Appendix A, we construct and analyze our PRG for block decision trees. In Appendix B, we give the proof of Corollary 4. In Appendix C, we explain our simulation of an oracle for **promise-BPP**, and in Appendix D, we suggest directions for further research. All other details can be found in the full version of this paper [10].

2 The shifting and rounding steward S_0

As a building block for our main steward constructions, we first construct our randomness-inefficient one-query steward S_0. Recall that any one-query steward makes two choices in each round: the input X_i to f_i and the estimate $Y_i \in \mathbb{R}^d$. The steward S_0 focuses on the second choice: each X_i is chosen uniformly at random, but S_0 carefully shifts and rounds the output $f_i(X_i)$. (See Figure 2.)

2.1 The shifting and rounding algorithm

We now describe the algorithm by which S_0 computes $Y_i \in \mathbb{R}^d$ from $W_i \stackrel{\text{def}}{=} f_i(X_i)$. Fix $n, k, d \in \mathbb{N}$ and $\varepsilon, \delta > 0$. Let $[d]$ denote the set $\{1, 2, \ldots, d\}$. Partition $\mathbb{R}$ into half-open intervals of length $(d+1) \cdot 2\varepsilon$. Let $\mathcal{I}$ denote the set of these intervals. For $w \in \mathbb{R}$, let $\mathsf{Round}(w)$ denote the midpoint of the interval in $\mathcal{I}$ containing w. Given $W_i \in \mathbb{R}^d$:

1. Find $\Delta_i \in [d+1]$ such that for every $j \in [d]$, there is some $I \in \mathcal{I}$ such that

$$[W_{ij} + (2\Delta_i - 1)\varepsilon, W_{ij} + (2\Delta_i + 1)\varepsilon] \subseteq I. \tag{6}$$

   (We will show that such a Δ_i exists.)
2. For every $j \in [d]$, set $Y_{ij} = \mathsf{Round}(W_{ij} + 2\Delta_i\varepsilon)$.

We must show that this algorithm is well-defined:

▶ **Lemma 7.** *For any $W \in \mathbb{R}^d$, there exists $\Delta \in [d+1]$ such that for every $j \in [d]$, there is a single interval in $\mathcal{I}$ that entirely contains $[W_j + (2\Delta - 1)\varepsilon, W_j + (2\Delta + 1)\varepsilon]$.*

Proof. Consider picking $\Delta \in [d+1]$ uniformly at random. For each j, the probability that two distinct intervals in $\mathcal{I}$ intersect $[W_j + (2\Delta - 1)\varepsilon, W_j + (2\Delta + 1)\varepsilon]$ is precisely $1/(d+1)$ by our choice of the length of the intervals. The union bound over d different j values completes the proof. ◀

2.2 Analysis: Certification trees

As outlined in Section 1.3.2, the key lemma says that for any owner O, there exists a block decision tree T_{O} with a small branching factor that certifies correctness of $\mathsf{O} \leftrightarrow \mathsf{S}_0$:

▶ **Lemma 8.** *Assume $\delta < 1/2$. Let $\Sigma = [d+1] \cup \{\perp\}$. For any deterministic owner O, there exists a (k, n, Σ) block decision tree T_{O} with the following properties.*

1. *For any internal node v, $\Pr_{X \in \{0,1\}^n}[v(X) = \perp] \le \delta$.*
2. *Fix $X_1, \dots, X_k \in \{0,1\}^n$, and suppose that the path from the root to $T_{\mathsf{O}}(X_1, \dots, X_k)$ does not include any $\perp$ nodes. Then $\max_i \|Y_i - \mu_i\|_\infty \le O(\varepsilon d)$ in $\mathsf{O} \leftrightarrow \mathsf{S}_0(X_1, \dots, X_k)$.*

Notice that Lemma 8 does not assert that T_{O} computes the transcript of $\mathsf{O} \leftrightarrow \mathsf{S}_0$. In fact, for the analysis, we will define *another* steward S'_0, and T_{O} will compute a sequence of values that arise in $\mathsf{O} \leftrightarrow \mathsf{S}'_0$. This new steward S'_0 will be computationally inefficient; it will *compress and decompress* the output of S_0 (with some chance of failure) before giving it to O, as we suggested in Section 1.3.2.

Proof of Lemma 8. For an (ε, δ)-concentrated function f, define $\mu(f)$ to be vector in $\mathbb{R}^d$ at which f is concentrated.[3] For a vector $Y \in \mathbb{R}^d$, say that a value $\Delta \in [d+1]$ is *f-compatible* with Y if $Y_j = \mathsf{Round}(\mu(f)_j + 2\Delta\varepsilon)$ for every $j \in [d]$. Just for the analysis, let S'_0 be the following (many-query) steward:

1. For $i = 1$ to k:
   a. Give f_i to S_0, allowing it to make its one query and choose its output vector $Y_i \in \mathbb{R}^d$.
   b. Query f_i at *every* point in its domain, thereby learning the entire function.
   c. Compute

$$\widehat{\Delta}_i = \begin{cases} \text{the smallest } \Delta \in [d+1] \text{ } f_i\text{-compatible with } Y_i & \text{if any such } \Delta \text{ exists} \\ \perp & \text{otherwise.} \end{cases} \tag{7}$$

   d. Output $\widehat{Y}_i = (\widehat{Y}_{i1}, \dots, \widehat{Y}_{id})$, where for each $j \in [d]$,

$$\widehat{Y}_{ij} = \begin{cases} \mathsf{Round}(\mu(f)_j + 2\widehat{\Delta}_i\varepsilon) & \text{if } \widehat{\Delta}_i \neq \perp \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

We are now ready to formally define T_{O} *as a function.* Because $\mathsf{S}'_0(X_1, \dots, X_k)$ looks at X_i only in round i, we can sensibly speak of the first i rounds of $\mathsf{O} \leftrightarrow \mathsf{S}'_0(X_1, \dots, X_i)$ even for $i < k$. This allows us to define $T_{\mathsf{O}}(X_1, \dots, X_i)$ to be the node v in T_{O} such that the path from the root to v is described by the values $\widehat{\Delta}_1, \dots, \widehat{\Delta}_i$ that arise in $\mathsf{O} \leftrightarrow \mathsf{S}'_0(X_1, \dots, X_i)$.

Now, we must show that this function T_{O} can be realized as a block decision tree, i.e. that each internal node v can be assigned a transition function $v : \{0,1\}^n \to \Sigma$ that is compatible with the definition of T_{O} as a function. Indeed, observe that $\widehat{\Delta}_1, \dots, \widehat{\Delta}_{i-1}$ fully determine the state of O after the first $i-1$ rounds of $\mathsf{O} \leftrightarrow \mathsf{S}'_0(X_1, \dots, X_i)$ and hence determine the function f_i. Furthermore, S_0 is "memoryless", i.e. Y_i is fully determined by f_i and X_i. Thus, $\widehat{\Delta}_i$ is fully determined by $\widehat{\Delta}_1, \dots, \widehat{\Delta}_{i-1}$ and X_i. So there is a function $\varphi : (\widehat{\Delta}_1, \dots, \widehat{\Delta}_{i-1}, X_i) \mapsto \widehat{\Delta}_i$, and if the path from the root to v is described by $\widehat{\Delta}_1, \dots, \widehat{\Delta}_{i-1}$, we can set $v(X_i) \stackrel{\text{def}}{=} \varphi(\widehat{\Delta}_1, \dots, \widehat{\Delta}_{i-1}, X_i)$.

[3] To handle non-uniqueness, let $\mu(f)$ be the *smallest* such vector under the lexicographic order. This exists, because $\{0,1\}^n$ is finite, so the set of points where f is concentrated is a compact subset of $\mathbb{R}^d$.

2.2.0.1 Analysis of T_{O}

By the definition of T_{O} as a function, to prove Item 1 in the lemma statement, we must show that in each round of $\mathsf{O} \leftrightarrow \mathsf{S}_0'$, $\Pr[\widehat{\Delta}_i = \perp] \leq \delta$. Indeed, by concentration, with probability $1-\delta$, for every j, $|W_{ij} - \mu(f_i)_j| \leq \varepsilon$. In this case, by the construction of S_0, $W_{ij} + 2\Delta_i\varepsilon$ and $\mu(f_i)_j + 2\Delta_i\varepsilon$ are in the same interval in $\mathcal{I}$ for every $j \in [d]$. Therefore, in this case, there is at least one Δ value that is f_i-compatible with Y_i, namely the value Δ_i used by S_0.

Finally, to prove Item 2 in the lemma statement, suppose the path from the root node to $T_{\mathsf{O}}(X_1, \ldots, X_k)$ does not include any $\perp$ nodes. Then in $\mathsf{O} \leftrightarrow \mathsf{S}_0'(X_1, \ldots, X_k)$, for every i, $\widehat{\Delta}_i \neq \perp$. This implies that every Y_{ij} is of the form $\mathsf{Round}(\mu(f_i)_j + 2\widehat{\Delta}_i\varepsilon)$ for some $\widehat{\Delta}_i \in [d+1]$. Therefore, $|Y_{ij} - \mu(f_i)_j| \leq 3(d+1)\varepsilon$, since $2\widehat{\Delta}_i\varepsilon \leq 2(d+1)\varepsilon$ and rounding introduces at most $(d+1)\varepsilon$ additional error.

Of course, so far the analysis has treated S_0', not S_0. But the crucial point is, for every i, since $\widehat{\Delta}_i \neq \perp$, we can be sure that $Y_i = \widehat{Y}_i$. Therefore, the values $f_1, \ldots, f_k, Y_1, \ldots, Y_k$ in $\mathsf{O} \leftrightarrow \mathsf{S}_0'(X_1, \ldots, X_k)$ are *exactly the same* as they are in $\mathsf{O} \leftrightarrow \mathsf{S}_0(X_1, \ldots, X_k)$! Therefore, in $\mathsf{O} \leftrightarrow \mathsf{S}_0(X_1, \ldots, X_k)$, for every i, $\|Y_i - \mu(f_i)\|_\infty \leq (3d+3)\varepsilon$. Finally, since $\delta < 1/2$, if μ_i is *any* point where f_i is (ε, δ)-concentrated, $\|\mu(f_i) - \mu_i\|_\infty \leq 2\varepsilon$. Therefore, for every i, $\|Y_i - \mu_i\|_\infty \leq 3(d+1)\varepsilon + 2\varepsilon = (3d+5)\varepsilon$. ◀

Notice that in $\mathsf{O} \leftrightarrow \mathsf{S}_0'(X_1, \ldots, X_k)$, if $\widehat{\Delta}_i = \perp$ for some i, then the interaction might diverge from $\mathsf{O} \leftrightarrow \mathsf{S}_0(X_1, \ldots, X_k)$, in which case $T_{\mathsf{O}}(X_1, \ldots, X_k)$ does not encode the transcript of $\mathsf{O} \leftrightarrow \mathsf{S}_0(X_1, \ldots, X_k)$ in any way.

3 Proof of main result (Theorem 3)

Without loss of generality, assume $\delta < 1/2$. (If $\delta \geq 1/2$, then either $k = 1$ or $k\delta \geq 1$; in either case, the result is trivial.) Let S_0 be the steward of Section 2, let Σ be the alphabet of Lemma 8, and let Gen be the γ-PRG for (k, n, Σ) block decision trees of Theorem 6. The steward is $\mathsf{S}(X) \stackrel{\text{def}}{=} \mathsf{S}_0(\mathsf{Gen}(X))$.

Consider any owner O. We may assume without loss of generality that O is deterministic, because a randomized owner is just a distribution over deterministic owners. By Item 1 of Lemma 8 and the union bound,

$$\Pr[\text{some node in the path from the root to } T_{\mathsf{O}}(U_{nk}) \text{ is labeled } \perp] \leq k\delta. \tag{9}$$

Therefore, when T_{O} reads $\mathsf{Gen}(U_s)$ instead of U_{nk}, the probability is at most $k\delta + \gamma$. By Item 2 of Lemma 8, this proves the correctness of S. The randomness complexity of S is just the seed length of Gen, which is indeed $n + O(k \log|\Sigma| + (\log k)\log(1/\gamma)) = n + O(k \log(d+1) + (\log k)\log(1/\gamma))$. The total runtime of S is clearly $\mathrm{poly}(n, k, d, \log(1/\varepsilon), \log(1/\gamma))$.[4] ◀

4 Application: the Goldreich-Levin algorithm

▶ **Theorem 9** (Randomness-efficient Goldreich-Levin algorithm). *There is a randomized algorithm that, given oracle access to $F : \{0,1\}^n \to \{-1,1\}$ and given input parameters $\delta, \theta > 0$, outputs a list L of subsets of $[n]$ such that with probability $1-\delta$,*

1. *every U satisfying $|\widehat{F}(U)| \geq \theta$ is in L, and*

[4] We assume here that our computational model allows the necessary arithmetic and rounding of Section 2.1 to be performed efficiently, even if the owner chooses an f_i that outputs vectors whose coordinates are very large numbers.

2. *every* $U \in L$ *satisfies* $|\widehat{F}(U)| \geq \theta/2$.

The number of queries made by the algorithm is $O\left(\frac{n}{\theta^{11}\log(1/\theta)}\log\left(\frac{n}{\delta\theta}\right)\right)$*, the number of random bits used by the algorithm is* $O(n+(\log n)\log(1/\delta))$*, and the runtime of the algorithm is* $\mathrm{poly}(n, 1/\theta, \log(1/\delta))$.

For comparison, using standard techniques, the Goldreich-Levin algorithm can be implemented in a straightforward way to use $O(\frac{n}{\theta^6}\log(\frac{n}{\delta\theta}))$ queries and $O(n^2 + n\log(\frac{n}{\delta\theta}))$ random bits. So our algorithm significantly improves the randomness complexity at the expense of substantially increasing the exponent of $1/\theta$ in the query complexity.

Toward proving Theorem 9, for a string $x \in \{0,1\}^{\leq n}$, define

$$\mathcal{U}(x) = \{U \subseteq [n] : \forall j \leq |x|, j \in U \iff x_j = 1\}. \tag{10}$$

(That is, we think of $x \in \{0,1\}^\ell$ as specifying $U \cap [\ell]$ in the natural way.) Define $W_x[F] = \sum_{U\in\mathcal{U}(x)} \widehat{F}(U)^2$. One of the key facts used in the standard Goldreich-Levin algorithm is that $W_x[F]$ can be estimated using few queries to F; here, we use the GW sampler to improve the randomness efficiency of that estimation.

▶ **Lemma 10.** *There is a randomized algorithm that, given oracle access to* F *and inputs* $x \in \{0,1\}^{\leq n}$*,* $\varepsilon, \delta > 0$*, estimates* $W_x[F]$ *to within* $\pm\varepsilon$ *with failure probability* δ*. The number of queries is* $O(\log(1/\delta)/\varepsilon^2)$*, the number of random bits is* $O(n + \log(1/\delta))$*, and the runtime is* $\mathrm{poly}(n, 1/\varepsilon, \log(1/\delta))$.

Proof. Let $\ell = |x|$. As shown in the proof of [19, Proposition 3.40],

$$W_x[F] = \mathop{\mathrm{E}}_{\substack{y,y'\in\{0,1\}^\ell \\ z\in\{0,1\}^{n-\ell}}}[F(y,z)\cdot F(y',z)\cdot\chi_x(y)\cdot\chi_x(y')], \tag{11}$$

where $\chi_x(y) \stackrel{\text{def}}{=} \prod_{j:x_j=1}(-1)^{y_j}$. Let $C : \{0,1\}^{n+\ell} \to \{0,1\}$ be the function

$$C(y,y',z) = \frac{1}{2} + \frac{1}{2}\cdot F(y,z)\cdot F(y',z)\cdot\chi_x(y)\cdot\chi_x(y'), \tag{12}$$

so that $W_x[F] = 2\,\mathrm{E}_{y,y',z}[C(y,y',z)] - 1$. We can estimate the expectation of C to within $\pm\varepsilon/2$ with failure probability δ using the GW sampler [9, Theorem 6.5], which implies an estimate of $W_x[F]$ to within $\pm\varepsilon$. The number of queries made by the GW sampler is $O(\log(1/\delta)/\varepsilon^2)$, and each query to C can be evaluated by making 2 queries to F. The randomness complexity of the GW sampler is $n + \ell + O(\log(1/\delta))$, which is $O(n + \log(1/\delta))$. ◀

The standard Goldreich-Levin algorithm proceeds by finding, for $\ell = 1$ to n, the set of all x with $|x| = \ell$ such that $W_x[F] \gtrsim \theta^2$. In each round, the algorithm estimates $W_x[F]$ for all strings x formed by appending a single bit to a string x' that was previously found to satisfy $W_{x'}[F] \gtrsim \theta^2$. This adaptive structure is exactly suited for saving random bits using a steward. To further drive down the randomness complexity, we reduce the number of rounds of adaptivity by appending $\log(1/\theta)$ bits at a time instead of 1 bit.

Proof of Theorem 9. Algorithm:

1. Let $u = \lfloor\log(1/\theta)\rfloor$, let $k = \lceil n/u\rceil$, and let $d = \lfloor 2^u \cdot 4/\theta^2\rfloor$.
2. Let S be a $(\theta^2/4, \delta)$-steward for k adaptively chosen $(\varepsilon, \delta/(2n))$-concentrated functions $f_1, \ldots, f_k : \{0,1\}^m \to \mathbb{R}^d$, where $\varepsilon \geq \Omega(\theta^2/d)$ and m will become clear later.
3. Set $L_0 := \{\text{empty string}\}$.
4. For $i = 1$ to k:

a. If $|L_{i-1}| > d/2^u$, abort and output "fail".
b. Observe that every string in L_{i-1} has length $\ell = u(i-1) < n$. Let $x_1, \ldots, x_t$ be the set of all strings obtained from strings in L_{i-1} by appending $\min\{u, n-\ell\}$ bits, so $t \leq 2^u |L_{i-1}| \leq d$.
c. Define $f_i : \{0,1\}^m \to \mathbb{R}^t$ by letting $f_i(X)_j$ be the estimate of $W_{x_j}[F]$ to within $\pm\varepsilon$ provided by the algorithm of Lemma 10 with failure probability $\delta/(2dn)$ using randomness X. Observe that by the union bound, f_i is $(\varepsilon, \delta/(2n))$-concentrated at $(W_{x_1}[F], \ldots, W_{x_t}[F])$.
d. By giving f_i to S, obtain estimates $\mu_1, \ldots, \mu_t$ for $W_{x_1}[F], \ldots, W_{x_t}[F]$.
e. Set $L_i := \{x_j : \mu_j \geq \theta^2/2\}$.

5. Output $L \stackrel{\text{def}}{=} \bigcup_{x \in L_k} \mathcal{U}(x)$.

(So m is the number of random bits used by the algorithm of Lemma 10.) With probability $1-\delta$, all of the responses of S are accurate, i.e. every μ_j value is within $\pm\theta^2/4$ of the corresponding $W_{x_j}[F]$ value. Assume from now on that this has happened.

By the definition of L_i, every x in every L_i satisfies $W_x[F] \geq \theta^2/4$. By Parseval's theorem (see, e.g., [19, Section 1.4]), this implies that $|L_i| \leq 4/\theta^2 \leq d/2^u$ for every i. Therefore, the algorithm does not abort. Let ℓ_i be the length of all the strings in L_i, so $\ell_i = ui$ for $i < k$ and $\ell_k = n$. Suppose $\widehat{F}(U)^2 \geq \theta^2$. By induction on i, the unique string $x \in \{0,1\}^{\ell_i}$ with $U \in \mathcal{U}(x)$ is placed in L_i, because the estimate of $W_x[F]$ is at least $3\theta^2/4 > \theta^2/2$. This shows that $U \in L$. Conversely, if U ends up in L, then the estimate of $\widehat{F}(U)^2$ in iteration $i = n$ was at least $\theta^2/2$, so $\widehat{F}(U)^2 \geq \theta^2/4$. This completes the proof of correctness of the algorithm.

Now, observe that the total number of queries to F is at most kd times the $O(\log(nd/\delta)/\varepsilon^2)$ queries that the algorithm of Lemma 10 makes, i.e. the total number of queries to F is

$$O\left(\frac{kd^3 \log(nd/\delta)}{\theta^2}\right) = O\left(\frac{n}{\theta^{11}\log(1/\theta)} \log\left(\frac{n}{\delta\theta}\right)\right).$$

The randomness complexity of the algorithm is just the randomness complexity of S. We will use the steward of Theorem 3 with $\gamma = \delta/2$, so the randomness complexity is $m + O(k\log(d+1) + (\log k)\log(1/\delta))$. Since $m \leq O(n + \log(n/(\delta\theta)))$, the total randomness complexity is

$$O\left(n + \frac{n}{\log(1/\theta)}\log(1/\theta) + (\log n)\log(1/\delta) + \log(1/\theta)\right) = O(n + (\log n)\log(1/\delta) + \log(1/\theta)).$$

To get rid of the $\log(1/\theta)$ term as claimed in the theorem statement, just notice that we can assume without loss of generality that $\theta \geq 2^{-n+1}$, because any nonzero Fourier coefficient of a $\{-1,1\}$-valued function has absolute value at least 2^{-n+1}. The total runtime of the algorithm is clearly $\text{poly}(n, 1/\theta, \log(1/\delta))$. ◀

References

1 R. Armoni. On the derandomization of space-bounded computations. In *Randomization and Approximation Techniques in Computer Science*, pages 47–59. Springer, 1998.
2 M. Bellare. A technique for upper bounding the spectral norm with applications to learning. In *Proceedings of the 5th Annual Workshop on Computational Learning Theory*, pages 62–70. ACM, 1992.
3 N. H. Bshouty, J. C. Jackson, and C. Tamon. More efficient PAC-learning of DNF with membership queries under the uniform distribution. *Journal of Computer and System Sciences*, 68(1):205–234, 2004.

4 H. Buhrman and L. Fortnow. One-sided versus two-sided error in probabilistic computation. In *Annual Symposium on Theoretical Aspects of Computer Science*, pages 100–109. Springer, 1999.

5 Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM journal on computing*, 38(1):97–139, 2008.

6 C. Dwork, V. Feldman, M. Hardt, T. Pitassi, O. Reingold, and A. L. Roth. Preserving statistical validity in adaptive data analysis. In *Proceedings of the 47th Annual Symposium on Theory of Computing*, STOC '15, pages 117–126. ACM, 2015.

7 E. Gat and S. Goldwasser. Probabilistic search algorithms with unique answers and their cryptographic applications. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 18, page 136, 2011.

8 O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*, STOC '89, pages 25–32. ACM, 1989.

9 O. Goldreich and A. Wigderson. Tiny families of functions with random properties: A quality-size trade-off for hashing. *Random Structures & Algorithms*, 11(4):315–343, 1997. doi:10.1002/(SICI)1098-2418(199712)11:4<315::AID-RSA3>3.0.CO;2-1.

10 W. M. Hoza and A. R. Klivans. Preserving randomness for adaptive algorithms. *arXiv preprint arXiv:1611.00783*, 2016.

11 R. Impagliazzo. *Pseudo-random generators for cryptography and for randomized algorithms*. PhD thesis, University of California, Berkeley, 1992. URL: http://cseweb.ucsd.edu/users/russell/format.ps.

12 R. Impagliazzo, N. Nisan, and A. Wigderson. Pseudorandomness for network algorithms. In *Proceedings of the 26th Annual Symposium on Theory of Computing*, STOC '94, pages 356–364. ACM, 1994.

13 R. Impagliazzo and D. Zuckerman. How to recycle random bits. In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science*, FOCS '89, pages 248–253. IEEE, 1989.

14 E. Kushilevitz and Y. Mansour. Learning decision trees using the Fourier spectrum. *SIAM Journal on Computing*, 22(6):1331–1348, 1993.

15 L. A. Levin. Randomness and nondeterminism. *Journal of Symbolic Logic*, 58(3):1102–1103, 1993.

16 P. Moser. Relative to *p promise-bpp* equals *app*. In *Electronic Colloquium on Computational Complexity (ECCC) Report TR01*, volume 68, 2001.

17 J. Naor and M. Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM journal on computing*, 22(4):838–856, 1993.

18 N. Nisan and D. Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–52, 1996.

19 R. O'Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.

20 R. Raz and O. Reingold. On recycling the randomness of states in space bounded computation. In *Proceedings of the 31st Annual ACM Symposium on Theory of Computing*, STOC '99, pages 159–168. ACM, 1999. doi:10.1145/301250.301294.

21 M. Saks and S. Zhou. $BP_HSPACE(S) \subseteq DSPACE(S^{3/2})$. *Journal of Computer and System Sciences*, 58(2):376–403, 1999.

22 S. P. Vadhan. *Pseudorandomness*, volume 56. Now, 2012.

A Pseudorandomness for block decision trees

In this section, we prove Theorem 6. Recall that our goal is to modify the internal parameters of the INW generator, thereby constructing a γ-PRG for (k, n, Σ) block decision trees with seed length $n + O(k \log |\Sigma| + (\log k) \log(1/\gamma)$. The construction and analysis mimic the standard treatment of the INW generator, and the reader who is familiar with the INW generator is encouraged to skip to Section A.4 to just see the new parameters. In words, the only new feature is that we use extractors for a geometrically growing entropy deficit at each level of the recursion to match the geometrically growing width of the block decision tree.

A.1 Formal definition of a PRG

Let U_n denote the uniform distribution on $\{0,1\}^n$. For two probability distributions μ, μ' on the same measurable space, write $\mu \sim_\gamma \mu'$ to indicate that μ and μ' have total variation distance at most γ.

▶ **Definition 11.** We say that $\mathsf{Gen} : \{0,1\}^s \to \{0,1\}^{nk}$ is a *γ-PRG for (k, n, Σ) block decision trees* if for every such tree T, $T(\mathsf{Gen}(U_s)) \sim_\gamma T(U_{nk})$.

A.2 Concatenating PRGs for block decision trees

Toward proving Theorem 6, for a (k, n, Σ) block decision tree $T = (V, E)$ and a node $v \in V$, let T_v denote the subtree rooted at v, and observe that we can think of T_v as a (k', n, Σ) block decision tree, where $k' = k - \text{depth}(v)$. This simple observation – after a block decision tree has been computing for a while, the remaining computation is just another block decision tree – implies that pseudorandom generators for block decision trees can be *concatenated* with mild error accumulation. This fact and its easy proof are perfectly analogous to the situation with ordered branching programs. We record the details below.

▶ **Lemma 12.** *Suppose* $\mathsf{Gen}_1 : \{0,1\}^{s_1} \to \{0,1\}^{nk_1}$ *is a γ_1-PRG for (k_1, n, Σ) block decision trees and* $\mathsf{Gen}_2 : \{0,1\}^{s_2} \to \{0,1\}^{nk_2}$ *is a γ_2-PRG for (k_2, n, Σ) block decision trees. Let* $\mathsf{Gen}(x, y) = (\mathsf{Gen}_1(x), \mathsf{Gen}_2(y))$. *Then* Gen *is a $(\gamma_1+\gamma_2)$-PRG for (k_1+k_2, n, Σ) block decision trees.*

Proof. Fix a $(k_1 + k_2, n, \Sigma)$ block decision tree T. For a node u at depth k_1 and a leaf node v, define

$$p(u) = \Pr[T(U_{nk_1}) = u] \qquad p(v \mid u) = \Pr[T_u(U_{nk_2}) = v]$$
$$\widetilde{p}(u) = \Pr[T(\mathsf{Gen}_1(U_{s_1})) = u] \qquad \widetilde{p}(v \mid u) = \Pr[T_u(\mathsf{Gen}_2(U_{s_2})) = v].$$

To prove correctness of Gen, recall that ℓ_1 distance is twice total variation distance. The ℓ_1 distance between $T(\mathsf{Gen}(U_{s_1+s_2}))$ and $T(U_{n(k_1+k_2)})$ is precisely $\sum_{u,v} |p(u)p(v \mid u) - \widetilde{p}(u)\widetilde{p}(v \mid u)|$. By the triangle inequality, this is bounded by

$$\begin{aligned}
&\sum_{u,v} |p(u)p(v \mid u) - p(u)\widetilde{p}(v \mid u)| + \sum_{u,v} |p(u)\widetilde{p}(v \mid u) - \widetilde{p}(u)\widetilde{p}(v \mid u)| \\
&= \sum_{u,v} p(u) \cdot |p(v \mid u) - \widetilde{p}(v \mid u)| + \sum_{u,v} |p(u) - \widetilde{p}(u)| \cdot \widetilde{p}(v \mid u) \\
&= \sum_u p(u) \sum_v |p(v \mid u) - \widetilde{p}(v \mid u)| + \sum_u |p(u) - \widetilde{p}(u)|.
\end{aligned}$$

By the correctness of Gen_1 and Gen_2, this is bounded by $(\sum_u p(u) \cdot 2\gamma_2) + 2\gamma_1 = 2(\gamma_1+\gamma_2)$. ◀

A.3 Recycling randomness

We find it most enlightening to think of the INW generator in terms of extractors, as suggested by Raz and Reingold [20] and in the spirit of the Nisan-Zuckerman generator [18]. The analysis is particularly clean if we work with *average-case extractors*, a concept introduced by Dodis et al. [5].

▶ **Definition 13.** For discrete random variables X, V, the *average-case conditional min-entropy* of X given V is

$$\widetilde{H}_\infty(X \mid V) = -\log_2 \left(\mathop{\mathrm{E}}_{v \sim V} \left[2^{-H_\infty(X|V=v)} \right] \right), \tag{13}$$

where H_∞ is (standard) min-entropy.

Intuitively, $\widetilde{H}_\infty(X \mid V)$ measures the amount of randomness in X from the perspective of someone who knows V. The output of an *average-case extractor* is required to look uniform even from the perspective of someone who knows V, as long as its first input is sampled from a distribution that has high min-entropy conditioned on V:

▶ **Definition 14.** We say that $\mathsf{Ext} : \{0,1\}^s \times \{0,1\}^d \to \{0,1\}^m$ is an *average-case* $(s-t, \beta)$*-extractor* if for every X distributed on $\{0,1\}^s$ and every discrete random variable V such that $\widetilde{H}_\infty(X \mid V) \geq s - t$, if we let $Y \sim U_d$ be independent of (X, V) and let $Z \sim U_m$ be independent of V, then $(V, \mathsf{Ext}(X, Y)) \sim_\beta (V, Z)$.

Average-case extractors are the perfect tools for *recycling randomness* in space-bounded computation. We record the details for block decision trees below.

▶ **Lemma 15** (Randomness recycling lemma for block decision trees)**.** *Suppose* $\mathsf{Gen} : \{0,1\}^s \to \{0,1\}^{nk}$ *is a* γ*-PRG for* (k, n, Σ) *block decision trees and* $\mathsf{Ext} : \{0,1\}^s \times \{0,1\}^d \to \{0,1\}^s$ *is an average-case* $(s - k \log |\Sigma|, \beta)$*-extractor. Define*

$$\mathsf{Gen}'(x, y) = (\mathsf{Gen}(x), \mathsf{Gen}(\mathsf{Ext}(x, y))). \tag{14}$$

Then Gen' *is a* $(2\gamma + \beta)$*-PRG for* $(2k, n, \Sigma)$ *block decision trees.*

Proof. Let T be a $(2k, n, \Sigma)$ block decision tree. Let $X \sim U_s$ and let $V = T(\mathsf{Gen}(X))$. By [5, Lemma 2.2b], the fact that V can be described using $k \log |\Sigma|$ bits implies that $\widetilde{H}_\infty(X \mid V) \geq s - k \log |\Sigma|$. Therefore, by the average-case extractor condition, if we let $Y \sim U_d$ be independent of X and $Z \sim U_d$ be independent of V, then

$$(V, \mathsf{Ext}(X, Y)) \sim_\beta (V, Z). \tag{15}$$

Applying a (deterministic) function can only make the distributions closer. Apply the function $(v, z) \mapsto T_v(\mathsf{Gen}(z))$:

$$T(\mathsf{Gen}'(X, Y)) \sim_\beta T(\mathsf{Gen}(X), \mathsf{Gen}(Z)). \tag{16}$$

By Lemma 12, the right-hand side is (2γ)-close to $T(U_{2nk})$. The triangle inequality completes the proof. ◀

To actually construct a generator, we will need to instantiate this randomness recycling lemma with an explicit average-case extractor:

▶ **Lemma 16.** *For every* $s, t \in \mathbb{N}$ *and every* $\beta > 0$*, there exists an average-case* $(s - t, \beta)$*-extractor* $\mathsf{Ext} : \{0,1\}^s \times \{0,1\}^d \to \{0,1\}^s$ *with seed length* $d \leq O(t + \log(1/\beta))$ *computable in time* $\mathrm{poly}(s, \log(1/\beta))$.

Proof sketch. It is standard (and can be proven using expanders, see, e.g., [22]) that there exists an *ordinary* $(s - t - \log(2/\beta), \beta/2)$-extractor $\mathsf{Ext} : \{0,1\}^s \times \{0,1\}^d \to \{0,1\}^s$ with seed length $d \leq O(t + \log(1/\beta))$ computable in time $\mathrm{poly}(s, \log(1/\beta))$. By the same argument as that used to prove [5, Lemma 2.3], Ext is automatically an average-case $(s-t, \beta)$-extractor. ◀

A.4 The recursive construction

Proof of Theorem 6. Define $\beta = \gamma/2^{\lceil \log k \rceil}$. For $i \geq 0$, define $s_i \in \mathbb{N}$, $d_i \in \mathbb{N}$, $G_i : \{0,1\}^{s_i} \to \{0,1\}^{n \cdot 2^i}$, and $\mathsf{Ext}_i : \{0,1\}^{s_i} \times \{0,1\}^{d_i} \to \{0,1\}^{s_i}$ through mutual recursion as follows. Start with $s_0 = n$ and $G_0(x) = x$. Having already defined s_i and G_i, let Ext_i be the average-case $(s_i - 2^i \log|\Sigma|, \beta)$-extractor of Lemma 16, and let d_i be its seed length. Then let $s_{i+1} = s_i + d_i$, and let

$$G_{i+1}(x, y) = (G_i(x), G_i(\mathsf{Ext}_i(x, y))). \tag{17}$$

We show by induction on i that G_i is a $(\beta \cdot (2^i - 1))$-PRG for $(2^i, n, \Sigma)$ block decision trees. In the base case $i = 0$, this is trivial. For the inductive step, apply Lemma 15, and note that $2\beta(2^i - 1) + \beta = \beta(2^{i+1} - 1)$. This completes the induction. Therefore, we can let $\mathsf{Gen} = G_{\lceil \log k \rceil}$, since $\beta \cdot (2^{\lceil \log k \rceil} - 1) < \gamma$. The seed length $s_{\lceil \log k \rceil}$ of Gen is

$$\begin{aligned} n + \sum_{i=0}^{\lceil \log k \rceil} d_i &\leq n + O\left(\sum_{i=0}^{\lceil \log k \rceil} (2^i \log|\Sigma| + \log k + \log(1/\gamma)) \right) \\ &\leq n + O(k \log|\Sigma| + (\log k)\log(1/\gamma)). \end{aligned}$$

The time needed to compute $\mathsf{Gen}(x)$ is just the time needed for $O(k)$ applications of Ext_i for various $i \leq O(\log k)$, which is $\mathrm{poly}(n, k, \log|\Sigma|, \log(1/\gamma))$. ◀

B Acceptance probabilities of Boolean circuits

In this section, we prove Corollary 4. To begin, we recall the definition of a *sampler*, which we used already in Section 4. A (ε, δ)-*sampler* for Boolean functions on n bits is a randomized oracle algorithm Samp such that for any Boolean function $C : \{0,1\}^n \to \{0,1\}$, if we let $\mu(C) \stackrel{\text{def}}{=} 2^{-n} \sum_x C(x)$, then

$$\Pr[|\mathsf{Samp}^C - \mu(C)| > \varepsilon] \leq \delta. \tag{18}$$

Proof of Corollary 4. Let c be the constant under the $O(\cdot)$ of the error ε' in the steward of Theorem 3. When given parameters $n, k, \varepsilon, \delta$, let Samp be the Boolean $(\varepsilon/c, \delta/(2k))$-sampler by Goldreich and Wigderson [9], and say it uses m coins. Let S be the (ε, δ)-steward of Theorem 3 for k adaptively chosen $(\varepsilon/c, \delta/(2k))$-concentrated functions $f_1, \ldots, f_k : \{0,1\}^m \to \mathbb{R}$. (So $\gamma = \delta/2$.) When given circuit C_i, define $f_i(X) = \mathsf{Samp}^{C_i}(X)$, i.e. the output of Samp^{C_i} with randomness X. Give f_i to S, and output the value Y_i that it returns.

Proof of correctness: The definition of a sampler implies that each f_i is $(\varepsilon/c, \delta/(2k))$-concentrated at $\mu(C_i)$. Furthermore, each f_i is defined purely in terms of C_i, which is chosen based only on $Y_1, \ldots, Y_{i-1}$. Therefore, the steward guarantee implies that with probability $1 - \delta$, every Y_i is within $\pm\varepsilon$ of $\mu(C_i)$.

Randomness complexity analysis: The number of bits m used by the sampler is $n + O(\log(k/\delta))$. Therefore, the number of bits used by the steward is

$$n + O(\log(k/\delta)) + O(k + (\log k)\log(1/\delta)) = n + O(k + (\log k)\log(1/\delta)). \tag{19}$$

Runtime analysis: The runtime of the steward is

$$\mathrm{poly}(m, k, \log(1/\gamma)) = \mathrm{poly}(n, k, \log(1/\delta)). \tag{20}$$

The runtime of the sampler is $\mathrm{poly}(n, 1/\varepsilon, \log k, \log(1/\delta))$. The time required to evaluate each query of the sampler in round i is $O(\mathrm{size}(C_i))$ (assuming we work with a suitable computational model and a suitable encoding of Boolean circuits.) The number of queries that the sampler makes in each round is $O(\log(k/\delta)/\varepsilon^2)$. Therefore, the total runtime of this algorithm is

$$O\left(\frac{\log k + \log(1/\delta)}{\varepsilon^2} \cdot \sum_{i=1}^{k} \mathrm{size}(C_i)\right) + \mathrm{poly}(n, k, 1/\varepsilon, \log(1/\delta)). \tag{21}$$

C Simulating an oracle for promise-BPP

Recall that **promise-BPP** is the class of promise problems that can be decided in probabilistic polynomial time with bounded failure probability. When an algorithm is given oracle access to a promise problem, it is allowed to make queries that violate the promise, and several models have been considered for dealing with such queries. Following Moser [16], we will stipulate that the oracle may respond in any arbitrary way to such queries. (See, e.g., [4] for two other models.) From these definitions, it is easy to show, for example, that $\mathbf{BPP}^{\mathbf{promise\text{-}BPP}} = \mathbf{BPP}$. In this section, using our steward, we give a time- and randomness-efficient simulation of any algorithm with an oracle for **promise-BPP**.

▶ **Theorem 17.** *Suppose a search problem* Π *can be solved by a deterministic* **promise-BPP**-*oracle algorithm that runs in time* T *and makes* k *queries, and suppose that (regardless of previous oracle responses) each query of this algorithm can be decided by a randomized algorithm that runs in time* T'*, uses* n *coins, and has failure probability* $1/3$*. Then for any* δ*,* Π *can be solved by a randomized (non-oracle) algorithm that runs in time*

$$T + O(T' \cdot k \log(k/\delta)) + \mathrm{poly}(n, k, \log(1/\delta)),$$

has randomness complexity

$$n + O(k + (\log k)\log(1/\delta)),$$

and has failure probability δ*.*

(Recall that search problems generalize decision problems and function problems. In reality, the theorem generalizes to just about any kind of "problem", but we restrict ourselves to search problems for concreteness.) The theorem can easily be extended to randomized oracle algorithms by considering the problem of executing the randomized oracle algorithm using a given randomness string.

Note that Theorem 17 would be trivial if it involved a **BPP** oracle instead of a **promise-BPP** oracle. Indeed, in the **BPP** case, the randomness can be reduced to just $n + O(\log k + \log(1/\delta))$. This is because a **BPP** algorithm is pseudodeterministic, so the randomness can be safely reused from one query to the next as discussed in Section 1.4. A **promise-BPP** algorithm is not pseudodeterministic in general – it is only guaranteed to be pseudodeterministic on inputs that satisfy the promise.

Proof sketch of Theorem 17. Let B be the algorithm of Corollary 4 with $\varepsilon = 1/10$ and the desired failure probability δ. When the oracle algorithm makes query i, define $f_i(X)$ to be the value outputted by the **promise-BPP** algorithm on that query string using randomness X. Give B the "circuit" f_i. (The algorithm B treats the circuits as black boxes, so we don't need to bother implementing f_i as a literal Boolean circuit; the important thing is that $f_i(X)$ can be evaluated in time T'.) When B outputs a value Y_i, give the oracle algorithm the response 0 if $Y_i < 1/2$ and 1 if $Y_i \geq 1/2$. ◀

D Directions for further research

The problem of randomness stewardship is fundamental, and the main open problem left by this work is to construct optimal stewards. The following are examples of concrete questions along these lines.

- Does every one-query steward with failure probability $\delta' \leq O(k\delta)$ have randomness complexity $n + \Omega(k \log(d+1))$? (Is the randomness complexity of our main steward near-optimal?)
- Does there exist a one-query $(O(\varepsilon), k\delta + 0.1)$-steward with randomness complexity $n + O(k \log(d+1))$? (Can the error of our main steward be improved?)

We explained in this work how the steward model captures some older derandomization constructions, and we gave new applications of stewards. We hope that future researchers find more connections and applications.

Commutative Algorithms Approximate the LLL-distribution

Fotis Iliopoulos[1]
University of California Berkeley, USA
fotis.iliopoulos@berkeley.edu
https://orcid.org/0000-0002-1825-0097

Abstract

Following the groundbreaking Moser-Tardos algorithm for the Lovász Local Lemma (LLL), a series of works have exploited a key ingredient of the original analysis, the witness tree lemma, in order to: derive deterministic, parallel and distributed algorithms for the LLL, to estimate the entropy of the output distribution, to partially avoid bad events, to deal with super-polynomially many bad events, and even to devise new algorithmic frameworks. Meanwhile, a parallel line of work has established tools for analyzing stochastic local search algorithms motivated by the LLL that do not fall within the Moser-Tardos framework. Unfortunately, the aforementioned results do not transfer to these more general settings. Mainly, this is because the witness tree lemma, provably, does not longer hold. Here we prove that for commutative algorithms, a class recently introduced by Kolmogorov and which captures the vast majority of LLL applications, the witness tree lemma does hold. Armed with this fact, we extend the main result of Haeupler, Saha, and Srinivasan to commutative algorithms, establishing that the output of such algorithms well-approximates the *LLL-distribution*, i.e., the distribution obtained by conditioning on all bad events being avoided, and give several new applications. For example, we show that the recent algorithm of Molloy for list coloring number of sparse, triangle-free graphs can output exponential many list colorings of the input graph.

2012 ACM Subject Classification Mathematics of computing → Probabilistic algorithms

Keywords and phrases Lovasz Local Lemma, Local Search, Commutativity, LLL-distribution, Coloring Triangle-free Graphs

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.44

Related Version A full version of the paper is available at `https://arxiv.org/abs/1704.02796`.

Acknowledgements The author is grateful to Dimitris Achlioptas and Alistair Sinclair for detailed comments and feedback, as well as to anonymous reviewers for comments and remarks.

1 Introduction

Many problems in combinatorics and computer science can be phrased as finding an object that lacks certain bad properties, or "flaws". In this paper we study algorithms that take as input a flawed object and try to remove all flaws by transforming the object through repeated probabilistic action.

Concretely, let Ω be a set of objects and let $F = \{f_1, f_2, \ldots, f_m\}$ be a collection of subsets of Ω. We will refer to each $f_i \in F$ as a *flaw* to express that its elements share some negative

[1] Research supported by NSF grant CCF-1514434 and the Onassis Foundation

feature. For example, if a CNF formula F on n variables has clauses $c_1, c_2, \ldots, c_m$, we can define for each clause c_i the flaw (subcube) $f_i \subseteq \{0,1\}^n$ whose elements violate c_i. Following linguistic rather than mathematical convention we say that f is present in σ if $f \ni \sigma$ and that $\sigma \in \Omega$ is *flawless* (perfect) if no flaw is present in σ.

To prove the *existence* of flawless objects we can often use the Probabilistic Method. As a matter of fact, in many interesting cases, this is the only way we know how to do so. To employ the Probabilistic Method, we introduce a probability measure μ over Ω and consider the collection of "bad" events corresponding to flaws. If we are able to show that the probability to avoid all bad events is strictly positive, then this implies the existence of a flawless object. A trivial example is the case where all the bad events are independent of one another and none of them has probability one. One of the most powerful tools of the Probabilistic Method is the Lovász Local Lemma [13] which weakens the latter restrictive condition of independence to a condition of limited dependence.

Making the LLL constructive was the study of intensive research for over two decades [5, 4, 27, 12, 33]. The breakthrough was made by Moser [28] who gave a very simple algorithm that finds a satisfying assignment of a k-CNF formula, under conditions that nearly match the LLL condition for satisfiability. Very shortly afterwards, Moser and Tardos [29] made the general LLL constructive for any *product* probability measure over explicitly presented variables. Specifically, they proved that whenever the general LLL conditon holds, the *Resample* algorithm, which repeatedly selects *any* occurring bad event and resamples all its variables according to the measure, i.e., independently, quickly converges to a flawless object.

The first result that made the LLL constructive in a non-product probability space was due to Harris and Srinivasan in [19], who considered the space of permutations endowed with the uniform measure. Subsequent works by Achlioptas and Iliopoulos [2, 1] introducing the flaws/actions framework, and of Harvey and Vondrák [21] introducing the resampling oracles framework, made the LLL constructive in more general settings. These frameworks [2, 21, 1] provide tools for analyzing *focused* stochastic search algorithms [30], i.e., algorithms which, like *Resample*, search by repeatedly selecting a flaw of the current state and moving to a random nearby state that avoids it, in the hope that, more often than not, more flaws are removed than introduced, so that a flawless object is eventually reached. At this point, all LLL applications we are aware of have efficient algorithms analyzable in these frameworks.

Besides conditions for existence and fast convergence to perfect objects, one could ask further questions regarding properties of focused search algorithms. For instance, "are they parallelizable ?", "how many solutions can they output?", "what is the expected "weight" of a solution?", etc. These questions and more have been answered for the Moser-Tardos algorithm in a long series of work [29, 15, 20, 22, 8, 11, 14, 17]. As a prominent example, the result of Haeupler, Saha and Srinivasan [15], as well as follow-up works of Harris and Srinivasan [20, 16], allow one to argue about the dynamics of the MT process, resulting in several new applications such as estimating the entropy of the output distribution, partially avoiding bad events, dealing with super-polynomially many bad events, and even new frameworks [18, 9].

Unfortunately, most of these follow-up results that further enhance, or exploit, our understanding of the MT process are not transferable to the general settings of [2, 21, 1]. Mainly, this is because a key and elegant technical result of the original analysis of Moser and Tardos, *the witness tree lemma*, does not longer hold under the most general assumptions [21]. Roughly, it states that any tree of bad events growing backwards in time from a certain root bad event A_i, with the children of each node A_j being bad events that are adjacent to A_j in the dependency graph, has probability of being consistent with the trajectory of

the algorithm that is bounded by the product of the probabilities of all events in this tree. The witness tree lemma and its variations [22, 14] has been used for several other purposes besides those already mentioned, such as designing deterministic, parallel and distributed algorithms for the LLL [29, 8, 11, 14, 17].

On the other hand, Harris and Srinivasan [19] do manage to prove the witness tree lemma for their algorithm for the LLL on the space of permutations, via an analysis that is tailored specifically to this setting. Although their proof does not seem to be easily generalizable to general spaces, their success makes it natural to ask if we can impose mild assumptions in the general settings of [2, 21, 1] under which the witness tree lemma (and most of its byproducts) can be established.

The main contribution of this paper is to answer this question positively by showing that it is possible to prove the witness tree lemma in the *commutative setting*. The latter was introduced by Kolmogorov [24], who showed that under its assumptions one can obtain parallel algorithms, as well as the flexibility of having arbitrary flaw choice strategy in the frameworks of [2, 21, 1]. We note that the commutative setting captures the vast majority of LLL applications, including but not limited to both the variable and the permutation settings.

Subsequently to the present work, Achlioptas, Iliopoulos and Sinclair [3] gave a simpler proof of the witness tree lemma under a more general notion of commutativity (essentially matrix commutativity) at the mild cost of slightly restricting the family of flaw choice strategies (as we will see, in this paper the flaw choice strategy can be arbitrary).

Armed with the witness tree lemma, we are able to study properties of algorithms in the commutative setting and give several applications.

Distributional Properties

As already mentioned, one of the most important applications of the witness tree lemma is given in the paper of Haeupler, Saha and Srinivasan [15], who study properties of the *MT-distribution*, the output distribution of the MT algorithm. Their main result is that the MT-distribution well-approximates the *LLL-distribution*, i.e., the distribution obtained by conditioning on all bad events being avoided. As an example, an immediate consequence of this fact is that one can argue about the expected *weight* of the output of the MT algorithm, given a weighting function over the space Ω. Furthermore, as shown in the same paper [15] and follow-up papers by Harris and Srinivasan [20, 16], one can lower bound the entropy of the MT distribution, go beyond the LLL conditions (if one is willing to only partially avoid bad events), and deal with applications with super-polynomially many number of bad events.

Here we extend the result of [15] to the commutative setting: Given a commutative algorithm that is perfectly compatible with the underlying probability measure, its output well-approximates the LLL distribution in the same sense the MT-distribution does in the variable setting. For arbitrary commutative algorithms, the quality of the approximation additionally depends on the compatibility of the algorithm with the measure on the event(s) of interest. A simplified, and imprecise, version of our main theorem, which assumes that the initial state of the algorithm is sampled according to the underlying probability distribution μ, is as follows. The formal statement of our main theorem can be found in Section 3.

▶ **Theorem 1** (Informal and Imprecise Statement). *If algorithm $\mathcal{A}$ is commutative and the algorithmic LLL conditions hold then, for each $E \subseteq \Omega$,*

$$\Pr[E] \leq \gamma(E) \left(1 + \frac{1}{d}\right)^{D_E} ,$$

where E is independent of all but at most D_E flaws, d is the maximum degree of the dependency graph, $\gamma(E) \geq \mu(E)$ is a measure of the "compatibility" between $\mathcal{A}$ and the underlying probability distribution μ at E, and $\Pr[E]$ is the probability that $\mathcal{A}$ ever reaches E during its execution.

Moreover, we quantitatively improve the bounds of [15] under the weaker assumptions of *Shearer's condition* [32], i.e., the most general LLL criterion under the assumption that the dependency graph is undirected. This allows us to study distributional properties of commutative algorithms using criteria that lie between the General LLL and Shearer's condition such as the Clique LLL [23].

Algorithmic LLL Without a Slack and Arbitrary Flaw Choice Strategy

The works of Achlioptas, Iliopoulos and Kolmogorov [2, 1, 24] require a multiplicative slack in the generalized LLL conditions in order to establish fast convergence to a perfect object. On the other hand, Harvey and Vondrák [21] dispense with this requirement in the important case of algorithms that are perfectly compatible with the underlying measure under the mild assumption that the dependency graph is undirected.

Using the witness lemma, we are able to dispense with the multiplicative slack requirement for arbitrary algorithms in the commutative setting and also have the flexibility of arbitrary flaw choice strategy, as in the result of Kolmogorov [24].

Improved Running Time Bounds

We are able to improve the running time bounds of Harvey and Vondrák [21] for commutative algorithms, matching those of Kolipaka and Szegedy [22] for the MT algorithm. Whether this could be done was left as an open question in [21]. We note that while the results of Achlioptas, Iliopoulos and Kolmogorov [2, 1, 24] also manage to give improved running time bounds they require a multiplicative-slack in the LLL conditions.

Concrete Applications

In the full version of the paper we give concrete applications of commutative algorithms showing new results for the problems of *rainbow matchings, list-coloring* and *acyclic edge coloring*. Each application is chosen so that it demonstrates specific features of our results. Perhaps the most interesting one, which we include in the present version, is to show that the algorithm of Molloy [26] for finding list-colorings in triangle-free graphs with maximum degree Δ using $(1+\epsilon)\frac{\Delta}{\ln \Delta}$ colors, can actually output exponentially many such colorings with positive probability. First, we show that Molloy's algorithm can be analyzed in the general frameworks of the algorithmic LLL and that it is commutative, a fact that gives us access to properties of its output distribution. Then, we apply results regarding the entropy of the output of commutative algorithms. We show the following theorem.

▶ **Theorem 2.** *For every $\epsilon > 0$ there exists Δ_ϵ such that every triangle-free graph G with maximum degree $\Delta \geq \Delta_\epsilon$ has list chromatic number $\chi_\ell(G) \leq (1+\epsilon)\frac{\Delta}{\ln \Delta}$. Moreover, there exists an algorithm $\mathcal{A}$ that takes G as input and list-colors it in expected polynomial time. In addition, $\mathcal{A}$ is able to output e^{cn} distinct list colorings with positive probability, where $c > 0$ is a constant that depends on ϵ and Δ.*

We emphasize that the algorithm of Molloy is a sophisticated stochastic local search algorithm whose analysis is far from any standard LLL setting. The fact that our results allow us to state non-trivial facts about its distributional properties almost in a black-box fashion is testament to their flexibility.

2 Background and Preliminaries

In this section we present the necessary background and definitions to describe our setting. In Subsection 2.1 we describe the Lovász Local Lemma. In Subsections 2.2 and 2.3 we formally outline the algorithmic assumptions of [2, 21, 1, 24]. In Subsection 2.4 we describe improved Lovász Local Lemma criteria formulated in our setting.

2.1 The Lovász Local Lemma

To prove the *existence* of flawless objects we can often use the Probabilistic Method. To do so, we introduce a probability measure μ over Ω and consider the collection of "bad" events corresponding to flaws. If we are able to show that the probability to avoid all bad events is strictly positive, then this implies the existence of a flawless object. One of the most powerful tools to establish the latter is the Lovász Local Lemma [13].

▶ **General LLL.** *Let (Ω, μ) be a probability space and $\mathcal{A} = \{A_1, A_2, \ldots, A_m\}$ be a set of m (bad) events. For each $i \in [m]$, let $D(i) \subseteq [m] \setminus \{i\}$ be such that $\mu(A_i \mid \cap_{j \in S} \overline{A_j}) = \mu(A_i)$ for every $S \subseteq [m] \setminus (D(i) \cup \{i\})$. If there exist positive real numbers $\{\psi_i\}_{i=1}^m$ such that for all $i \in [m]$,*

$$\frac{\mu(A_i)}{\psi_i} \sum_{S \subseteq D(i) \cup \{i\}} \prod_{j \in S} \psi_j \leq 1 \ , \tag{1}$$

then the probability that none of the events in $\mathcal{A}$ occurs is at least $\prod_{i=1}^m 1/(1+\psi_i) > 0$.

▶ Remark. Condition (1) above is equivalent to the more well-known form $\mu(A_i) \leq x_i \prod_{j \in D(i)}(1 - x_j)$, where $x_i = \psi_i/(1+\psi_i)$. As we will see, formulation (1) facilitates refinements.

Let G be the digraph over the vertex set $[m]$ with an edge from each $i \in [m]$ to each element of $D(i) \cup \{i\}$. We call such a graph a *dependency* graph. Therefore, at a high level, the LLL states that if there exists a sparse dependency graph and each bad event is not too likely, then perfect objects exist.

2.2 Algorithmic Framework

Here we describe the class of algorithms we will consider as well as the algorithmic LLL criteria for fast convergence to a perfect object. Since we will be interested in algorithms that search for perfect objects, we sometimes refer to Ω as a state space and to its elements as states.

For a state σ, we denote by $U(\sigma) = \{j \in [m] \text{ s.t. } f_j \ni \sigma\}$ the set of indices of flaws that are present at σ. We consider algorithms which at each flawed state σ choose an element of $U(\sigma)$ and randomly move to a nearby state in an effort to *address* the corresponding flaw. Concretely, we will assume that for every flaw f_i and every state $\sigma \in f_i$ there is a probability distribution $\rho_i(\sigma, \cdot)$ with a non-empty support $A(i, \sigma) \subseteq \Omega$ such that addressing flaw f_i at state σ amounts to selecting the next state σ' from $A(i, \sigma)$ with probability $\rho_i(\sigma, \sigma')$. We call $A(i, \sigma)$ the set of *actions* for addressing flaw f_i at σ and note that potentially $A(i, \sigma) \cap f_i \neq \emptyset$, i.e., addressing a flaw does not necessarily imply removing it. The actions for flaw f_i form a digraph D_i on Ω having an arc $\sigma \xrightarrow{i} \sigma'$ for each pair $(\sigma, \sigma') \in f_i \times A(i, \sigma)$. Let D be the multi-digraph on Ω that is the union of all D_i.

We consider algorithms that start from a state $\sigma \in \Omega$ picked from an initial distribution θ, and then repeatedly pick a flaw that is present in the current state and address it. The algorithm always terminates when it encounters a flawless state.

To state the algorithmic LLL criteria for fast convergence of such algorithms we need to introduce two key ingredients. The first one is a notion of *causality* among flaws that will be used to induce a graph over $[m]$, which will play a role similar to the one of the dependency graph in the existential Local Lemma formulation. We note that there is a formal connection between causality graphs and dependency graphs (for more details see [21]).

▶ **Causality.** *For an arc $\sigma \xrightarrow{i} \sigma'$ in D_i and a flaw f_j present in σ' we say that f_i causes f_j if $f_i = f_j$ or $f_j \not\ni \sigma$. If D_i contains* any *arc in which f_i causes f_j we say that f_i* potentially causes *f_j.*

▶ **Causality Digraph.** *Any digraph $C = C(\Omega, F, D)$ on $[m]$ where $i \to j$ exists whenever f_i potentially causes f_j is called a* causality digraph. *The* neighborhood *of a flaw f_i in C is $\Gamma(i) = \{j : i \to j \text{ exists in } C\}$.*

The second ingredient is a measure of *compatibility* between the actions of the algorithm for addressing each flaw f_i (that is, digraph D_i) and the probability measure μ over Ω which we will use for the analysis. As was shown in [21, 1, 24] one can capture compatibility by letting

$$d_i = \max_{\sigma \in \Omega} \frac{\nu_i(\sigma)}{\mu(\sigma)} \geq 1 \ , \tag{2}$$

where $\nu_i(\sigma)$ is the probability of ending up at state σ at the end of the following experiment: sample $\omega \in f_i$ according to μ and address flaw f_i at ω. An algorithm achieving perfect compatibility for flaw f_i, i.e., $d_i = 1$, is a *resampling oracle* for flaw f_i (observe that the Moser-Tardos algorithm is trivially a resampling oracle for every flaw). More generally, ascribing to each flaw f_i the *charge*

$$\gamma(f_i) = d_i \cdot \mu(f_i) = \max_{\sigma' \in \Omega} \frac{1}{\mu(\sigma')} \sum_{\sigma \in f_i} \mu(\sigma) \rho_i(\sigma, \sigma') \ ,$$

yields the following algorithmization condition. If for every flaw $f_i \in F$,

$$\frac{\gamma(f_i)}{\psi_i} \sum_{S \subseteq \Gamma(i)} \prod_{j \in S} \psi_j < 1 \tag{3}$$

then there exists a flaw choice strategy under which the algorithm will reach a perfect object fast. (In most applications, that is in $O\left(\log|\Omega| + m \log_2\left(\frac{1+\psi_{\max}}{\psi_{\min}}\right)\right)$ steps with high probability.)

Throughout the paper we assume that we are given an undirected causality graph C (and thus the relation $\Gamma(\cdot)$ is symmetric) and we will sometimes write $i \sim j$ if $j \in \Gamma(i) \leftrightarrow j \in \Gamma(j)$. Furthermore, for a set $S \subseteq [m]$ we define $\Gamma(S) = \bigcup_{i \in S} \Gamma(i)$. Finally, we denote by $\mathrm{Ind}(S) = \mathrm{Ind}_C(S)$ the set of independent subsets of S with respect to C.

2.3 Commutativity

We will say that $\sigma \xrightarrow{i} \sigma'$ is a *valid trajectory* if it is possible to get from state σ to state σ' by addressing flaw f_i as described in the algorithm, i.e., if two conditions hold: $i \in U(\sigma)$ and $\sigma' \in A(i, \sigma)$. Kolmogorov [24] described the following *commutativity* condition. We call the setting in which Definition 3 holds *the commutative setting.*

▶ **Definition 3** (Commutativity [24]). A tuple $(F, \sim, \rho)$ is called *commutative* if there exists a mapping Swap that sends any trajectory $\Sigma = \sigma_1 \xrightarrow{i} \sigma_2 \xrightarrow{j} \sigma_3$ with $i \nsim j$ to another valid trajectory $\text{Swap}(\Sigma) = \sigma_1 \xrightarrow{j} \sigma_2' \xrightarrow{i} \sigma_3$, and:

1. Swap is injective,
2. $\rho_i(\sigma_1, \sigma_2)\rho_j(\sigma_2, \sigma_3) = \rho_j(\sigma_1, \sigma_2')\rho_i(\sigma_2', \sigma_3)$.

It is straightforward to check that the Moser Tardos algorithm satisfies the commutativity condition. Furthermore, Kolmogorov showed that the same is true for resampling oracles in the permutation [19] and perfect matchings [21] settings, and Harris [17] designed commutative resampling oracles for hamiltonian cycles.

Finally, as already mentioned, Kolmogorov showed that in the commutativity setting one may choose an arbitrary flaw choice strategy which is a function of the entire past execution history. The same will be true for our results, so we make the convention that given a tuple $(F, \sim, \rho)$ we always fix some arbitrary flaw choice strategy to get a well-defined, commutative algorithm $\mathcal{A} = (F, \sim, \rho)$.

2.4 Improved LLL Criteria

Besides the general form of the LLL (1) there exist improved criteria that apply in the full generality of the LLL setting. The most well-known are the *cluster expansion condition* [7] and the *Shearer's condition* [32]. Both of these criteria apply when the dependency graph is undirected and have been made constructive [22, 31, 2, 21, 1, 24] in the most general algorithmic LLL settings.

Cluster Expansion Condition

The cluster expansion criterion strictly improves upon the General LLL criterion (1) by taking advantage of the local density of the dependency graph.

▶ **Definition 4.** Given a sequence of positive real numbers $\{\psi_i\}_{i=1}^m$, we say that the cluster expansion condition is satisfied if for each $i \in [m]$:

$$\frac{\gamma(f_i)}{\psi_i} \sum_{S \in \text{Ind}(\Gamma(i))} \prod_{j \in S} \psi_j \leq 1 \ . \tag{4}$$

Shearer's Condition

Let $\gamma \in \mathbb{R}^m$ be the real vector such that $\gamma_i = \gamma(f_i)$. Furthermore, for $S \subseteq [m]$ define $\gamma_S = \prod_{j \in S} \gamma_j$ and the polynomial q_S:

$$q_S = q_S(\gamma) = \sum_{\substack{I \in \text{Ind}([m]) \\ S \subseteq I}} (-1)^{|I|-|S|} \gamma_I \ .$$

▶ **Definition 5.** We say that the Shearer's condition is satisfied if $q_S(\gamma) \geq 0$ for all $S \subseteq [m]$, and $q_\emptyset(\gamma) > 0$.

3 Statement of Results

Assuming that the LLL conditions (1) hold, the *LLL-distribution*, which we denote by μ_{LLL}, is defined as the distribution induced by the measure μ conditional on no bad event occurring. The following proposition relates the LLL distribution to measure μ making it a powerful

tool that can be used to argue about properties of flawless objects. The idea is that if an (not necessarily bad) event E is independent from most bad events, then its probability under the LLL distribution is not much larger than its probability under the probability measure μ.

▶ **Proposition 6** ([15]). *If the LLL conditions* (1) *hold, then for any event E:*

$$\mu_{\mathrm{LLL}}(E) \leq \mu(E) \sum_{S \subseteq D(E)} \prod_{j \in S} \psi_j \ , \tag{5}$$

where $D(E) \subseteq [m]$ is such that $\mu(E \mid \bigcap_{j \in S} \overline{A}_j) = \mu(E)$ for all $S \subseteq [m] \setminus D(E)$.

The main result of Haeupler, Saha and Srinivasan [15] is that the Moser-Tardos algorithm approximates well the LLL distribution, in the sense that the left-hanside of (5) bounds the probability that it ever reaches a subspace $E \subseteq \Omega$ during its execution. Building on this fact, [15] and followup works [20, 16] manage to show several new applications.

Here we extend the latter result to arbitrary commutative algorithms. Given an arbitrary set $E \subseteq \Omega$ and a commutative algorithm $\mathcal{A}$, consider an extension, $\mathcal{A}_E$, of $\mathcal{A}$ by defining an extra flaw $f_{m+1} \equiv E$ with its own set of probability distributions $\rho_{m+1}(\sigma, \cdot), \sigma \in E$. If $\mathcal{A}$ is commutative with respect to $\sim$, we will say that $\mathcal{A}_E$ is a *commutative extension* of $\mathcal{A}$ if $\mathcal{A}_E = (F \cup \{m+1\}, \sim, \rho)$ is also commutative.

Commutative extensions should be interpreted as a tool to bound the probability that $\mathcal{A}$ ever reaches a subset E of the state space. That is, they are defined only for the purposes of the analysis and, typically in applications, they are a natural extension of the algorithm. For example, in the case of the Moser-Tardos algorithm applied to k-SAT, if one would like to bound the probability that the algorithm ever reaches a state such that variables x_1, x_2 of the formula are both set to true, then one could define $f_{m+1} = \{\sigma \in \Omega \text{ s.t. } \sigma(x_1) = \sigma(x_2) = 1\}$ along with the corresponding commutative extension of the Moser-Tardos algorithm that addresses f_{m+1} by resampling variables x_1, x_2 according to the product measure over the variables of the formula that the Moser-Tardos algorithm uses whenever it needs to resample a violated clause. Indeed, commutative extensions of this form are implicitly defined in the analysis of [15] for the Moser-Tardos algorithm.

We will use the notation $\Pr[\cdot] = \Pr_{\mathcal{A}}[\cdot]$ to refer to the probability of events in the probability space induced by the execution of algorithm $\mathcal{A}$. For example, the probability that $\mathcal{A}$ ever reaches a set $E \subseteq \Omega$ of the state space during its execution will be denoted by $\Pr[E]$.

▶ **Theorem 7.** *If $\mathcal{A} = (F, \sim, \rho)$ is commutative and the cluster expansion condition is satisfied then:*

1. *for each $i \in [m]$: $\mathbb{E}[N_i] \leq \lambda_{\mathrm{init}} \psi_i$;*
2. *for each $E \subseteq \Omega$: $\Pr[E] \leq \lambda_{\mathrm{init}} \gamma(E) \sum_{S \in \mathrm{Ind}(\Gamma(E))} \prod_{j \in S} \psi_j$;*

where N_i is the number of times flaw f_i is addressed during the execution of $\mathcal{A}$, $\lambda_{\mathrm{init}} = \max_{\sigma \in \Omega} \frac{\theta(\sigma)}{\mu(\sigma)}$, and $\Gamma(E)$ and $\gamma(E)$ are defined with respect to a fixed commutative extension $\mathcal{A}_E$.

▶ **Corollary 8.** *Algorithm $\mathcal{A}$ terminates after $O(\lambda_{\mathrm{init}} \sum_{i \in [m]} \psi_i)$ steps in expectation.*

▶ Remark. If the Shearer's condition is satisfied, then one can replace ψ_i in Theorem 7 with $\frac{q_{\{i\}}(\gamma)}{q_{\emptyset}(\gamma)}$.

We note that the first part of Theorem 7 allows us to guarantee fast convergence of $\mathcal{A}$ to a perfect object without having to assume a "slack" in the cluster expansion and Shearer's conditions (unlike the works of [1, 24]) and, moreover, improves upon the (roughly quadratically worse) running bound of [21], matching the one of [22]. Whether the latter could be done was left as an open question in [21].

3.1 Entropy of the Output Distribution

Here we present one of the main byproducts of Theorem 7. The reader is referred to the full version of the paper for applications of Theorem 7 in partially avoiding flaws and dealing with settings with super-polynomially many flaws.

An elegant application of the known bounds for the Moser-Tardos distribution is estimating its randomness. In particular, Harris and Srinivasan [20] show that one can give lower bounds on the *Rényi entropy* of the output of the Moser-Tardos algorithm.

▶ **Definition 9** ([10]). Let ν be a probability measure over a finite set S. The *Rényi entropy with parameter* ρ of ν is defined to be

$$H_\rho[\nu] = \frac{1}{1-\rho} \ln \sum_{s \in S} \nu(s)^\rho \ .$$

The *min-entropy* H_∞ is a special case defined as $H_\infty[\nu] = \lim_{\rho \to \infty} H_\rho[\nu] = -\ln \max_{v \in S} \nu(\sigma)$.

Using the results of Section 3 we can show the analogous result in our setting.

▶ **Theorem 10.** *Assume that $\mathcal{A} = (F, \sim, \rho)$ is commutative, the cluster expansion condition is satisfied. Let ν be the output distribution of $\mathcal{A}$. Then, for $\rho > 1$,*

$$H_\rho[\nu] \geq H_\rho[\mu] - \frac{\rho}{\rho - 1} \ln \left(\sum_{S \in \mathrm{Ind}([m])} \prod_{j \in S} \psi_j \right) - \frac{\rho}{\rho - 1} \ln \lambda_{\mathrm{init}} \ .$$

Given Theorem 7, the proof is akin to the analogous result in [20] and can be found in the full version of the paper.

▶ Remark. Using Shearer's condition we can replace ψ_i with $\frac{q_{\{i\}}(\gamma)}{q_\emptyset(\gamma)}$, $i \in [m]$.

A straightforward application of having a lower bound on $H_\rho[\nu]$ (for any ρ), where ν is the output distribution of the algorithm, is that there exist at least $\exp(H_\rho[\nu])$ flawless objects. Before [20], the authors in [25] also used the (existential) LLL for enumeration of combinatorial structures by exploiting the fact that it guarantees a small probability p of avoiding all flaws when sampling from the uniform measure (and, thus, their number is at least $p|\Omega|$).

4 List-Coloring of Triangle-Free Graphs

In the problem of list coloring one is given a graph $G = G(V, E)$ over n vertices $V = \{v_1, v_2, \ldots, v_n\}$ and, for each $v \in V$, a list of colors $\mathcal{L}_v$. The goal is to find a list coloring $\sigma \in \mathcal{L}_{v_1} \times \ldots \times \mathcal{L}_{v_n}$ of G such that $\sigma(v) \neq \sigma(u)$ for any pair of adjacent vertices.

The *list chromatic number* $\chi_\ell(G)$ of a graph G is the minimum number of colors for which such a coloring is attainable. A celebrated result of Johansson shows that there exist a large constant $C > 0$ such that every triangle-free graph with maximum degree $\Delta \geq \Delta_0$ can be properly list-colored using $C\Delta/\ln \Delta$ colors. Very recently, Molloy [26] improved Johansson's result showing that C can be replaced by $(1 + \epsilon)$ for any $\epsilon > 0$ assuming that $\Delta \geq \Delta_\epsilon$.(We note that, soon after, Bernshteyn [6] established the same bound for the list chromatic number using the LLL. However, his result is not constructive as it uses a sophisticated probability measure for which it is not clear how one could design "efficient" resampling oracles.)

Here we show how that the algorithm of Molloy is amenable to our analysis and, in particular, we prove that it can output exponentially many proper colorings with positive probability.

4.1 The Algorithm

The algorithm of [26] works in two stages. First, it finds a partial list-coloring which has the property that (i) each vertex $v \in V$ has "many"' available colors; (ii) there is not "too much competition" for the available colors of v, i.e., they do not appear in the list of available colors of its neighbors. Then, it completes the coloring via a fairly straightforward application of the Moser-Tardos algorithm.

To describe the algorithm formally, we will need some further notation. First, it will be convenient to treat Blank as a color that is in the list of every vertex. For each vertex v and *partial* list-coloring σ let

- N_v denote the set of vertices adjacent to v;
- $L_v(\sigma) \subseteq \mathcal{L}_v$ to be the set of *available* colors for v at state σ, i.e., the set of colors that we can assign to v in σ without making any edge monochromatic. Notice that Blank is always an available color;
- $T_{v,c}(\sigma)$ to be the set of vertices $u \in N_v$ such that $\sigma(u) = \text{Blank}$ and $c \in L_u(\sigma)$.

Let $\Omega = \prod_{v \in V} \mathcal{L}_v$ and define $L = \Delta^{\frac{\epsilon}{2}}$. Given a partial list-coloring, we define the following flaws for any vertex v:

$$B_v = \{\sigma \in \Omega : |L_v(\sigma)| < L\};$$

$$Z_v = \left\{\sigma \in \Omega : \sum_{c \in L_v(\sigma) \setminus \text{Blank}} |T_{v,c}(\sigma)| > \frac{1}{10} L \cdot |L_v(\sigma)|\right\} .$$

▶ **Lemma 11** (The Second Phase). *Given a flawless partial list coloring, a complete list-coloring of G can be found in expected polynomial time.*

Lemma 11 was proved in [26] via a fairly straightforward application of the Lovász Local Lemma, and can be made constructive via the Moser-Tardos algorithm. What is left is to describe the first phase of the algorithm.

- The initial distribution θ, which is important in this case, is chosen to be the following: Fix an independent set S of G of size at least $n/(\Delta + 1)$. (This is trivial to find efficiently via a greedy algorithm). Choose one color from $\mathcal{L}_u \setminus \text{Blank}$, $u \in S$, uniformly at random, and assign it to u;
- to address a flaw $f \in \{B_v, Z_v\}$ at state σ, for each $u \in N_v$, choose uniformly at random a color from $L_u(\sigma)$ and assign it to u;
- as a flaw choice strategy, the algorithm first fixes any ordering π over flaws. At every step, it chooses the lowest occurring flaw according to π and addresses it.

4.2 Proving Termination

Let $\mathcal{A}_1$, $\mathcal{A}_2$ denote the first and second phase of our algorithm, respectively. Here we prove that $\mathcal{A}_1$ terminates in expected polynomial time. To do so, we will use the convergence result corresponding to equation (3). (Although, as we will see, $\mathcal{A}_1$ is commutative for an appropriate choice of a causality graph, we won't use Theorem 7 to prove its convergence. This is because λ_{init} is exponentially large in this case).

The measure μ we use for the analysis is the the uniform measure over proper list-colorings. We will use the following lemma whose proof can be found in Section B of the Appendix.

▶ **Lemma 12.** *For each vertex v and flaw $f \in \{B_v, Z_v\}$ we have that*

$$\gamma(f) \leq 2\Delta^{-4} .$$

Consider the causality graph such that $f_v \sim f_u$, if $\text{dist}(u,v) \leq 3$, where f_v, f_u are either B-flaws or Z-flaws. Notice that it has maximum degree at most $2(\Delta^3+1)$. Setting $\psi_f = \psi = \frac{1}{2(\Delta^3+1)}$ for any flaw f and applying (3), we get that the algorithm converges in expected polynomial time since

$$\gamma(f) \sum_{S \subseteq \Gamma(f)} \prod_{g \in S} \psi_g \leq \frac{2}{\Delta^4} \cdot 2(\Delta^3+1) \cdot \mathrm{e} < 4\mathrm{e}\left(\frac{1}{\Delta} + \frac{1}{\Delta^4}\right) < 1 \ ,$$

for large enough Δ for any flaw $f \in \{B_v, Z_v\}$ and $\log_2 |\Omega| + m \log_2 \left(\frac{1+\psi}{\psi}\right) = O(n \log n + m \log_2 n)$.

4.3 A Lower Bound on the Number of Possible Outputs

The bound regarding the number of list colorings the algorithm can output with positive probability follows almost immediately from the two following lemmata.

▶ **Lemma 13.** *Algorithm $\mathcal{A}_1$ can output at least $\exp\left(n\left(\frac{\ln q}{\Delta+1} - \frac{1}{\Delta^3}\right)\right)$ flawless partial list-colorings with positive probability.*

Proof. It is not hard to verify that $\mathcal{A}_1 = (F, \sim, \rho)$ is commutative. Applying Theorem 10 we get that $\mathcal{A}_1$ can output at least

$$\exp\left(\ln \frac{|\Omega|}{\lambda_{\text{init}}} - \sum_{f \in F} \psi_f\right) > \exp\left(n\left(\frac{\ln q}{\Delta+1} - \frac{1}{\Delta^3}\right)\right) \ , \tag{6}$$

flawless partial colorings. ◀

▶ **Lemma 14.** *Suppose $\mathcal{A}_1$ can output N flawless partial list-colorings with positive probability. Suppose further that among these partial list-colorings, the ones with the lowest number of colored vertices have exactly αn vertices colored, $\alpha \in (0,1)$. Then, $\mathcal{A}_2$ can output at least $\max\left(N 2^{-(1-\alpha)n}, \left(\frac{8L}{11}\right)^{(1-\alpha)n}\right)$ list-colorings with positive probability.*

The proof of Lemma 14 can be found in the full version of the paper.

Proof of Theorem 2. Combining Lemmata 13, 14 we get that our algorithm outputs at least

$$\exp\left(n \min_{\alpha} \max\left(f(\Delta) - (1-\alpha)\ln 2, (1-\alpha)\ln\left(\frac{8L}{11}\right)\right)\right) = \mathrm{e}^{cn} \ ,$$

list-colorings with positive probability, where $f(\Delta) = \frac{\ln q}{\Delta+1} - \frac{1}{\Delta^3}$ and $c = f(\Delta)\frac{\ln \frac{8L}{11}}{\ln \frac{16L}{11}}$, concluding the proof. ◀

References

1 Dimitris Achlioptas and Fotis Iliopoulos. Focused stochastic local search and the Lovász local lemma. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 2024–2038, 2016. doi:10.1137/1.9781611974331.ch141.

2 Dimitris Achlioptas and Fotis Iliopoulos. Random walks that find perfect objects and the Lovász local lemma. *J. ACM*, 63(3):22:1–22:29, 2016. doi:10.1145/2818352.

3 Dimitris Achlioptas, Fotis Iliopoulos, and Alistair Sinclair. A new perspective on stochastic local search and the lovasz local lemma. *CoRR*, abs/1805.02026, 2018. arXiv:1805.02026.

4 Noga Alon. A parallel algorithmic version of the local lemma. *Random Struct. Algorithms*, 2(4):367–378, 1991. doi:10.1002/rsa.3240020403.

5 József Beck. An algorithmic approach to the Lovász local lemma. I. *Random Structures Algorithms*, 2(4):343–365, 1991. doi:10.1002/rsa.3240020402.

6 Anton Bernshteyn. The johansson–molloy theorem for dp-coloring. *arXiv preprint arXiv:1708.03843*, 2017.

7 Rodrigo Bissacot, Roberto Fernández, Aldo Procacci, and Benedetto Scoppola. An improvement of the Lovász local lemma via cluster expansion. *Combinatorics, Probability & Computing*, 20(5):709–719, 2011. doi:10.1017/S0963548311000253.

8 Karthekeyan Chandrasekaran, Navin Goyal, and Bernhard Haeupler. Deterministic algorithms for the Lovász local lemma. *SIAM J. Comput.*, 42(6):2132–2155, 2013. doi:10.1137/100799642.

9 Antares Chen, David G. Harris, and Aravind Srinivasan. Partial resampling to approximate covering integer programs. In Robert Krauthgamer, editor, *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1984–2003. SIAM, 2016. doi:10.1137/1.9781611974331.ch139.

10 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.

11 Kai-Min Chung, Seth Pettie, and Hsin-Hao Su. Distributed algorithms for the Lovász local lemma and graph coloring. In Magnús M. Halldórsson and Shlomi Dolev, editors, *ACM Symposium on Principles of Distributed Computing, PODC '14, Paris, France, July 15-18, 2014*, pages 134–143. ACM, 2014. doi:10.1145/2611462.2611465.

12 Artur Czumaj and Christian Scheideler. Coloring non-uniform hypergraphs: a new algorithmic approach to the general Lovász local lemma. In *Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms (San Francisco, CA, 2000)*, pages 30–39, 2000.

13 Paul Erdős and László Lovász. Problems and results on 3-chromatic hypergraphs and some related questions. In *Infinite and finite sets (Colloq., Keszthely, 1973; dedicated to P. Erdős on his 60th birthday), Vol. II*, pages 609–627. Colloq. Math. Soc. János Bolyai, Vol. 10. North-Holland, Amsterdam, 1975.

14 Bernhard Haeupler and David G Harris. Parallel algorithms and concentration bounds for the Lovász local lemma via witness-DAGs. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1170–1187. SIAM, 2017.

15 Bernhard Haeupler, Barna Saha, and Aravind Srinivasan. New constructive aspects of the Lovász local lemma. *J. ACM*, 58(6):Art. 28, 28, 2011. doi:10.1145/2049697.2049702.

16 David G. Harris. New bounds for the Moser-Tardos distribution: Beyond the Lovasz local lemma. *CoRR*, abs/1610.09653, 2016. arXiv:1610.09653.

17 David G. Harris. Oblivious resampling oracles and parallel algorithms for the lopsided Lovász local lemma. *CoRR*, abs/1702.02547, 2017. arXiv:1702.02547.

18 David G. Harris and Aravind Srinivasan. The Moser-Tardos framework with partial resampling. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 469–478. IEEE Computer Society, 2013. doi:10.1109/FOCS.2013.57.

19 David G. Harris and Aravind Srinivasan. A constructive algorithm for the Lovász local lemma on permutations. In Chandra Chekuri, editor, *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 907–925. SIAM, 2014. doi:10.1137/1.9781611973402.68.

20 David G. Harris and Aravind Srinivasan. Algorithmic and enumerative aspects of the Moser-Tardos distribution. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 2004–2023, 2016. doi:10.1137/1.9781611974331.ch140.

21 Nicholas J. A. Harvey and Jan Vondrák. An algorithmic proof of the Lovász local lemma via resampling oracles. In Venkatesan Guruswami, editor, *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 1327–1346. IEEE Computer Society, 2015. doi:10.1109/FOCS.2015.85.

22 Kashyap Babu Rao Kolipaka and Mario Szegedy. Moser and Tardos meet Lovász. In *STOC*, pages 235–244. ACM, 2011. doi:10.1145/1993636.1993669.

23 Kashyap Babu Rao Kolipaka, Mario Szegedy, and Yixin Xu. A sharper local lemma with improved applications. In Anupam Gupta, Klaus Jansen, José D. P. Rolim, and Rocco A. Servedio, editors, *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*, volume 7408 of *Lecture Notes in Computer Science*, pages 603–614. Springer, 2012. doi:10.1007/978-3-642-32512-0_51.

24 Vladimir Kolmogorov. Commutativity in the algorithmic Lovász local lemma. In Irit Dinur, editor, *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 780–787. IEEE Computer Society, 2016. doi:10.1109/FOCS.2016.88.

25 Linyuan Lu and Laszlo A Szekely. A new asymptotic enumeration technique: the Lovász local lemma. *arXiv preprint arXiv:0905.3983*, 2009.

26 Michael Molloy. The list chromatic number of graphs with small clique number. *arXiv preprint arXiv:1701.09133*, 2017.

27 Michael Molloy and Bruce Reed. Further algorithmic aspects of the local lemma. In *STOC '98 (Dallas, TX)*, pages 524–529. ACM, New York, 1999.

28 Robin A. Moser. A constructive proof of the Lovász local lemma. In *STOC'09—Proceedings of the 2009 ACM International Symposium on Theory of Computing*, pages 343–350. ACM, New York, 2009.

29 Robin A. Moser and Gábor Tardos. A constructive proof of the general Lovász local lemma. *J. ACM*, 57(2):Art. 11, 15, 2010. doi:10.1145/1667053.1667060.

30 Christos H. Papadimitriou. On selecting a satisfying truth assignment. In *FOCS*, pages 163–169. IEEE Computer Society, 1991. doi:10.1109/SFCS.1991.185365.

31 Wesley Pegden. An extension of the Moser-Tardos algorithmic local lemma. *SIAM J. Discrete Math.*, 28(2):911–917, 2014. doi:10.1137/110828290.

32 J.B. Shearer. On a problem of Spencer. *Combinatorica*, 5(3):241–245, 1985. doi:10.1007/BF02579368.

33 Aravind Srinivasan. Improved algorithmic versions of the Lovász local lemma. In Shang-Hua Teng, editor, *SODA*, pages 611–620. SIAM, 2008. URL: http://dl.acm.org/citation.cfm?id=1347082.1347150.

A Proof of Main Results

In this section we state and prove the witness tree lemma for our setting. We then use it to prove Theorem 7. Some of the proofs are omitted for the sake of brevity, but all of them can be found in the full version of the paper.

A.1 The Witness Tree Lemma

Given a trajectory $\Sigma = \sigma_1 \xrightarrow{w_1} \dots \sigma_t \xrightarrow{w_t} \sigma_{t+1}$ we denote by $W(\Sigma) = (w_1, \dots, w_t)$ the *witness sequence* of Σ. (Recall that according to our notation, w_i denotes the index of the flaw that was addressed at the i-th step).

To state the witness tree lemma, we will first need to recall the definition of witness trees from [29], slightly reformulated to fit our setting. A witness tree $\tau = (T, \ell_T)$ is a finite rooted, unordered, tree T along with a labelling $\ell_T : V(T) \to [m]$ of its vertices with indices of flaws such that the children of a vertex $v \in V(T)$ receives labels from $\Gamma(\ell(v))$. To lighten the notation, we will sometimes write $[v]$ to denote $\ell(v)$ and $V(\tau)$ instead of $V(T)$. Given a witness sequence $W = (w_1, w_2, \dots, w_t)$ we associate with each $i \in [t]$ a witness tree $\tau_W(i)$ that is constructed as follows: Let $\tau_W^{(i)}(i)$ be an isolated vertex labelled by w_i. Then, going backwards for each $j = i-1, i-2, \dots, 1$: if there is a vertex $v \in \tau_W^{j+1}(i)$ such that $[v] \sim w_j$ then we choose among those vertices the one having the maximum distance (breaking ties arbitrarily) from the root and attach a new child vertex u to v that we label w_j to get $\tau_W^{(j)}(i)$. If there is no such vertex v then $\tau_W^{(j+1)}(i) = \tau_W^{(j)}(i)$. Finally, let $\tau_W(i) = \tau_W^{(1)}(i)$.

We will say that a witness tree τ occurs in a trajectory Σ if $W(\Sigma) = (w_1, w_2, \dots, w_t)$ and there is $k \in [t]$ such that $\tau_W(k) = \tau$.

▶ **Theorem 15** (The witness tree lemma). *Assume that $\mathcal{A} = (F, \sim, \rho)$ is commutative. Then, for every witness tree τ we have that:*

$$\Pr[\tau] \leq \lambda_{\text{init}} \prod_{v \in V(\tau)} \gamma(f_{[v]}) \ .$$

We show the proof of Theorem 15 in Section A.4.

A.2 Witness Trees and Stable Witness Sequences

Here we show some properties of witness trees (which are induced by witness sequences of the algorithm) that will be useful to us later. We also draw a connection between witness trees and *stable witness sequences*, which we will need in the proof of Theorem 7. Stable witness sequences were first introduced in [22] to make the Shearer's criterion constructive in the variable setting.

A.2.1 Properties of Witness Trees

The following propositions capture the main properties of witness trees we will need.

▶ **Proposition 16.** *For a witness tree $\tau = (T, \ell_T)$ let $L_i = L_i(\tau)$ denote the set of labels of the nodes at distance i from the root. For each $i \geq 0$, $L_i \in \text{Ind}([m])$.*

▶ **Proposition 17.** *For a witness sequence W of length t and any two distinct $i, j \in [t]$ we have that $\tau_W(i) \neq \tau_W(j)$.*

A.2.2 Stable Witness Sequences

We will now recall the definition of *stable sequences*. Variations of this notion have also been used by [21, 24] to make the Shearer's criterion constructive in the more general settings of the algorithmic Local Lemma. Here we will use the following definition:

▶ **Definition 18.** A sequence of subsets $(I_1, \ldots, I_k)$ of $[m]$ with $k \geq 1$ is called *stable* if
1. $I_r \in \mathrm{Ind}([m]) \setminus \{\emptyset\}$ for each $r \in [k]$
2. $I_{r+1} \subseteq \Gamma(I_r)$ for each $r \in [k-1]$.

▶ **Definition 19.** A witness sequence $W = (w_1, \ldots, w_t)$ is called *stable* if it can be partitioned into non-empty sequences as $W = (W_1, \ldots, W_k)$ such that the elements of each sequence W_r are distinct, and the sequence $\phi_W := (I_1, \ldots, I_k)$ is stable, where I_r is the set of indices of flaws in W_r (for $r \in [k]$).

For any arbitrary ordering π among indices of flaws, if in addition each sequence $W_r = (w_i, \ldots, w_j)$ satisfies $w_i \prec_\pi \ldots \prec_\pi w_j$ then W is called π- *stable*.

▶ **Proposition 20.** *([24]) For a stable witness sequence the partitioning in Definition 19 is unique.*

For a witness sequence $W = (w_1, \ldots, w_t)$ let $\mathrm{Rev}[W] = (w_t, \ldots, w_1)$ denote the reverse sequence. Let also R_W denote the first set (the "root") of the stable sequence $\phi_W := (I_1, \ldots, I_k)$, i.e., $R_W = I_1$. Finally, let $\mathcal{R}_i^\pi$ be the set of witness sequences W such that $\mathrm{Rev}[W]$ is π-stable and $R_{\mathrm{Rev}[W]} = \{i\}$.

There is a connection between stable sequences and witness trees that we will need for the proofs of Theorem 7 and which we will describe below.

Let $\mathcal{W}_i$ denote the set of witness trees with root labelled by i. For each $\tau \in \mathcal{W}_i$, let $\chi_\pi(\tau)$ be the *ordered* witness tree that is induced by ordering the children of each node in τ from left to right, increasingly according to π. Define $\mathcal{W}_i^\pi := \chi_\pi(\mathcal{W}_i)$ and observe that χ_π is a bijection. Finally, recall that for a witness tree τ we denote by $L_j(\tau)$ the set of labels of the nodes at distance j from the root.

▶ **Lemma 21.** *There is a bijection χ_i^π mapping $\mathcal{R}_i^\pi$ to $\mathcal{W}_i^\pi$ with the following property: Let $W \in \mathcal{R}_i^\pi$ and let $(I_1 = \{i\}, I_2, \ldots, I_k)$ be the unique partitioning of* $\mathrm{Rev}[W]$ *guaranteed by Proposition 20. Then, $I_j = L_{j-1}\left(\chi_i^\pi(W)\right)$ for each $j \in [k]$.*

A.2.3 Counting Witness Trees

In our proofs we will need to bound the sum over all trees $\tau \in \mathcal{W}_i$ of the product of charges of the (labels of) the nodes of each tree τ. Fortunately, the method for doing that is well trodden by now (see for example [29, 31]). Recall that $\mathcal{W}_i$ denotes the set of all possible witness trees with root that is labelled by i.

▶ **Lemma 22.** *If the Cluster Expansion condition is satisfied then:*

$$\sum_{\tau \in \mathcal{W}_i} \prod_{v \in V(\tau)} \gamma\left(f_{[v]}\right) \leq \psi_i \ .$$

We also show the following lemma that can be used whenever the Shearer's condition applies.

▶ **Lemma 23.** *If the Shearer's condition is satisfied then:*

$$\sum_{\tau \in \mathcal{W}_i} \prod_{v \in V(\tau)} \gamma\left(f_{[v]}\right) \leq \frac{q_{\{i\}}(\gamma)}{q_\emptyset(\gamma)} \ .$$

▶ Remark. We note that if we have assumed a stronger "cluster expansion condition" (namely, in (4) we have $\Gamma(i) \cup \{i\}$) instead of $\Gamma(i)$ then Corollary 22 could have also been shown as an immediate application of Lemma 23, since it is known ([7, 21, 24]) that, in this case, for every $i \in [m]$ we have that $\frac{q_{\{i\}}(\gamma)}{q_{\emptyset}(\gamma)} \leq \psi_i$.

A.3 Proof of Theorem 7

We first prove Theorem 7. The first part follows by Theorem 15, Lemma 22 and Proposition 17 that suggest:

$$\mathbb{E}[N_i] \leq \lambda_{\text{init}} \sum_{\tau \in \mathcal{W}_i} \prod_{v \in V(\tau)} \gamma(f_{[v]}) \leq \lambda_{\text{init}} \psi_i \ .$$

To see the second part of Theorem 7, consider the new set of flaws $F' = F \cup \{f_{m+1}\}$, where $f_{m+1} = E$, as well as a "truncated" commutative extension $\mathcal{A}'$ of $\mathcal{A}$ with the following properties:

- **(i)** For each state $\sigma \notin f_{m+1}$ algorithm $\mathcal{A}'$ invokes $\mathcal{A}$ to choose the next state.
- **(ii)** $\gamma(E) := \gamma_{\mathcal{A}'}(f_{m+1})$.
- **(iii)** f_{m+1} is always of the highest priority: when at a state $\sigma \in f_{m+1}$, $\mathcal{A}'$ chooses to address f_{m+1}.
- **(iv)** $\mathcal{A}'$ stops after it addresses f_{m+1} for the first time.

By coupling $\mathcal{A}$ and $\mathcal{A}'$ we see that $\Pr_{\mathcal{A}}[E] = \Pr_{\mathcal{A}'}[f_{m+1}]$. Let $\mathcal{W}_E$ be the set of witness trees that might occur in an execution of $\mathcal{A}'$ and whose root is labelled by $m+1$. Notice that due to property (iv) of $\mathcal{A}'$ every tree $\tau \in \mathcal{W}_E$ contains exactly one node (the root) labelled by $m+1$, while every other node is labelled by elements in $[m]$. Furthermore, the set of labels of the children of the root of τ is an element of $\text{Ind}(\Gamma(E))$. Finally, if v is a node that corresponds to a child of the root in τ, then the subtree τ_v that is rooted at v is an element of $\mathcal{W}_{[v]}$. Using Theorem 15 and the fact that $\mathcal{A}'$ is commutative we can bound $\Pr_{\mathcal{A}}[E]$ as

$$\sum_{\tau \in \mathcal{W}_E} \Pr_{\mathcal{A}'}[\tau] \leq \lambda_{\text{init}} \gamma(E) \sum_{S \in \text{Ind}(\Gamma(E))} \left(\prod_{j \in S} \sum_{\tau \in \mathcal{W}_j} \prod_{v \in \tau} \gamma([v]) \right) \leq \lambda_{\text{init}} \gamma(E) \sum_{S \in \text{Ind}(\Gamma(E))} \prod_{j \in S} \psi_j,$$

where the last equality follows from Corollary 22. The proof of Theorem 7 in the Shearer's condition regime is identical, where instead of Lemma 22 we use Lemma 23.

A.4 Proof of Lemma 15

Throughout the proof, we will use ideas and definitions from [24]. We also note that we will assume w.l.o.g. that algorithm $\mathcal{A}$ follows a deterministic flaw choice strategy. This is because randomized flaw choice strategies can equivalently be interpreted as convex combination of deterministic ones (and therefore, randomized strategies can be seen as taking expectation over deterministic ones).

For a trajectory Σ of length t we define

$$p(\Sigma) = \lambda_{\text{init}} \prod_{i=1}^{t} \rho_{w_i}(\sigma_i, \sigma_{i+1})$$

and notice that $\Pr[\Sigma] \leq p(\Sigma)$. Furthermore, we say that a trajectory Σ' is a proper prefix of Σ if Σ' is a prefix of Σ and $\Sigma \neq \Sigma'$.

▶ **Definition 24** ([24]). A set $\mathcal{X}$ of trajectories of the algorithm will be called *valid* if (i) all trajectories in $\mathcal{X}$ follow the same deterministic flaw choice strategy (not necessarily the same used by $\mathcal{A}$), and (ii) for any $\Sigma, \Sigma' \in \mathcal{X}$ trajectory Σ is not a proper prefix of Σ'.

▶ **Lemma 25** ([24]). *Consider a witness sequence $W = (w_1, \ldots, w_t)$ and a valid set of trajectories $\mathcal{X}$ such that W is a prefix of $W(\Sigma)$ for every $\Sigma \in \mathcal{X}$. Then:*

$$\sum_{\Sigma \in \mathcal{X}} p(\Sigma) \leq \lambda_{\text{init}} \prod_{i=1}^{t} \gamma(f_{w_i}) \ ,$$

A *swap* is the operation of transforming a trajectory $\Sigma = \ldots \sigma_1 \xrightarrow{i} \sigma_2 \xrightarrow{j} \sigma_3 \ldots$, with $i \nsim j$, to a trajectory $\Sigma' = \ldots \sigma_1 \xrightarrow{j} \sigma_2' \xrightarrow{i} \sigma_3 \ldots$, where $\sigma_1 \xrightarrow{j} \sigma_2' \xrightarrow{i} \sigma_3 = \text{Swap}(\sigma_1 \xrightarrow{i} \sigma_2 \xrightarrow{j} \sigma_3)$. A mapping Φ on a set of trajectories will be called a *swapping mapping* if it operates by applying a sequence of swaps.

The main idea now will be to construct a swapping mapping whose goal will be to transform trajectories of the algorithm to a form that satisfies certain properties by applying swaps .

For a trajectory Σ in which a tree $\tau \in \mathcal{W}_i$ occurs, we denote by W_Σ^τ the prefix of $W(\Sigma)$ up to the step that corresponds to the root of τ (observe that Proposition 17 mandates that there exists a unique such step). Notice that, since $\tau \in \mathcal{W}_i$, the algorithm addresses flaw f_i at this step, and thus the final element of W_Σ^τ is $\{i\}$. Finally, recall the definitions of $\mathcal{R}_i^\pi$, χ_π and χ_i^π.

▶ **Lemma 26.** *Fix a witness tree $\tau \in \mathcal{W}_i$ and let $\mathcal{X}^\tau$ be a valid set of trajectories in which τ occurs. If $\mathcal{A} = (F, \sim, \rho)$ is commutative then there exists a set of trajectories $\mathcal{X}_\pi^\tau$ and a swapping mapping $\Phi^\tau : \mathcal{X}^\tau \to \mathcal{X}_\pi^\tau$ which is a bijection such that*
(a) for any $\Sigma \in \mathcal{X}_\pi^\tau$ we have that W_Σ^τ is the unique witness sequence in $\mathcal{R}_i^\pi$ such that $\chi_i^\pi(W_\Sigma^\tau) = \chi_\pi(\tau)$;
(b) for any witness sequence W the set $\{\Sigma \in \mathcal{X}_\pi^\tau \mid \text{Rev}[W_\Sigma^\tau] = W\}$ is valid.

We prove Lemma 26 in Section A.5. To see Theorem 15, consider a witness tree $\tau \in \mathcal{W}_i$, and let $\mathcal{Y}^\tau$ be the set of all trajectories that $\mathcal{A}$ may follow in which τ occurs. Now remove from $\mathcal{Y}^\tau$ any trajectory Σ for which there exists a trajectory Σ' such that Σ is a proper prefix of Σ' to get $\mathcal{X}^\tau$. Clearly, this is a valid set and so recalling that χ_π is a bijection and applying Lemma 26 we have that:

$$\Pr[\tau] = \sum_{\Sigma \in \mathcal{X}^\tau} \Pr[\Sigma] \leq \sum_{\Sigma \in \mathcal{X}^\tau} p(\Sigma) = \sum_{\Sigma \in \mathcal{X}_\pi^\tau} p(\Sigma) \ , \tag{7}$$

where to get the second equality we use the second requirement of Definition 3. Lemma 26 further implies that for every trajectory $\Sigma \in \mathcal{X}_\pi^\tau$ we have that W_Σ^τ is the (unique) witness sequence in $\mathcal{R}_i^\pi$ such that $\chi_i^\pi(W_\Sigma^\tau) = \chi_\pi(\tau)$, i.e., $W_\Sigma^\tau = (\chi_i^\pi)^{-1}(\chi_\pi(\tau))$. This means that the witnesses of the trajectories in $\mathcal{X}_\pi^\tau$ have $W := (\chi_i^\pi)^{-1}(\chi_\pi(\tau))$ as a common prefix. Since part (b) of Lemma 26 implies that $\mathcal{X}_\pi^\tau$ is valid, applying Lemma 25 we get:

$$\sum_{\Sigma \in \mathcal{X}_\pi^\tau} p(\Sigma) \leq \lambda_{\text{init}} \prod_{w \in W} \gamma(f_w) = \lambda_{\text{init}} \prod_{v \in V(\tau)} \gamma(f_{[v]}) \quad , \tag{8}$$

where the second inequality follows from the fact that $\chi_i^\pi(W) = \chi_\pi(\tau)$ and $V(\tau) = V(\chi_\pi(\tau))$, concluding the proof.

A.5 Proof of Lemma 26

Our proof builds on the proof of Theorem 19 in [24]. We will be denoting witness sequences $W = (w_1, w_2, \ldots, w_t)$ as a sequence of *named indices of flaws* $W = (\mathbf{w}_1, \ldots, \mathbf{w}_t)$ where $\mathbf{w}_j = (w_j, n_j)$ and $n_j = |\{k \in [j] \mid w_k = w_j\}| \geq 1$ is the number of occurrences of w_j in the length-j prefix of W. Note that a named index $\mathbf{w}$ cannot appear twice in a sequence W. Finally, if $\mathbf{w}$ is a named index of flaw we denote by w (that is, without bold font) the flaw index that is associated with it.

For a trajectory Σ such that $W(\Sigma) = (\mathbf{w}_1, \ldots, \mathbf{w}_t)$ we define a directed acyclic graph $\mathbf{G}(\Sigma) = (\mathbf{V}(\Sigma), \mathbf{E}(\Sigma))$ where $\mathbf{V}(\Sigma) = \{\mathbf{w}_1, \ldots, \mathbf{w}_t\}$ and $\mathbf{E}(\Sigma) = \{(\mathbf{w}_j, \mathbf{w}_k)$ s.t. $w_j \sim w_k$ and $j < k$ $\}$. This means that we have an edge from a named flaw $\mathbf{w}_i$ to another flaw $\mathbf{w}_j$ whenever their corresponding flaw indices are related according to $\sim$ and $\mathbf{w}_j$ occurs in Σ before $\mathbf{w}_k$.

By Proposition 17, for any trajectory Σ in which τ occurs there is a unique step $t^* = t^*(\Sigma)$ such that $\tau_{W(\Sigma)}(t^*) = \tau$. For such a trajectory Σ, let $\mathbf{Q}(\Sigma) \subseteq \mathbf{V}(\Sigma)$ be the set of flaws from which the node $\mathbf{w}_{t^*}$ can be reached in $\mathbf{G}(\Sigma)$, where $\mathbf{w}_{t^*}$ is the named flaw index that corresponds to the step t^*. Notice that $w_{t^*} = i$ (since $\tau \in \mathcal{W}_i$). For $\mathbf{w} \in \mathbf{Q}(\Sigma)$ let $d(\mathbf{w})$ be the length of the longest path from $\mathbf{w}$ to $\mathbf{w}_{t^*}$ in $\mathbf{G}(\Sigma)$ plus one. For example, $d(\mathbf{w}_{t^*}) = 1$.

Let $Q(\Sigma)$ denote the sequence consisting of the named flaws in $\mathbf{Q}(\Sigma)$ listed in the order they appear in Σ. The idea is to repeatedly apply the operation Swap to Σ so that we reach a trajectory Σ' that has a permutation $Q_\pi(\Sigma)$ of $Q(\Sigma)$ as a prefix. In particular, we will show that $Q_\pi(\Sigma) \in \mathcal{R}_i^\pi$ and $\chi_i^\pi(Q_\pi(\Sigma)) = \chi_\pi(\tau)$.

To that end, for an integer $r \geq 1$ define $\mathbf{I}_r = \{\mathbf{w} \in \mathbf{Q}(\Sigma) \mid d(\mathbf{w}) = r\}$, and let Q_r be the sequence consisting of the named flaw indices in $\mathbf{I}_r$ sorted in decreasing order with respect to π. Then, we define $Q_\pi(\Sigma) = (Q_s, \ldots, Q_1)$ where $s = \max\{d(\mathbf{w}) \mid \mathbf{w} \in \mathbf{Q}(\Sigma)\}$.

▶ **Lemma 27.** *$Q_\pi(\Sigma) \in \mathcal{R}_i^\pi$ and $\chi_i^\pi(Q_\pi(\Sigma)) = \chi_\pi(\tau)$.*

Proof. Let $Y = Y(\Sigma) = \mathrm{Rev}[Q_\pi(\Sigma)] = (Q_1, Q_2, \ldots, Q_s)$ be the reverse sequence of Σ. By definition, $R_Y = Q_1 = \{i\}$. To show that $Q \in \mathcal{R}_i^\pi$ it suffices to show that $Q_{i+1} \subseteq \Gamma(Q_i)$ for each $i \in [s-1]$. To see this, recall the definitions of $\mathbf{I}_{r+1}$ and Q_{r+1} and observe that, for each $i_{r+1} \in Q_{r+1}$, there must be a path of r indices of flaws $i_r, i_{r-1}, \ldots, i_1$ such that for every $j \in [r-1]$ we have that $i_j \in Q_j$ and $i_j \sim i_{j+1}$.

Let k be the number of elements in witness sequence $Q(\Sigma)$. Recall that $\chi_\pi^i(Q_\pi(\Sigma)) := \chi_\pi(\tau_{Q_\pi(\Sigma)}(k))$ (proof of Lemma 21). The proof is concluded by also recalling the algorithm for constructing witness trees and observing that $\tau_{Q_\pi(\Sigma)}(k) = \tau_{W(\Sigma)}(t^*) = \tau$. ◀

Note that applying Swap to Σ does not affect graph $\mathbf{G}(\Sigma)$ and set $\mathbf{Q}(\Sigma)$ and, thus, neither the sequence $Q_\pi(\Sigma)$. With that in mind, we show next how we could apply Swap repeatedly to $\Sigma \in \mathcal{X}^\tau$ to reach a Σ' such $Q_\pi(\Sigma)$ is a prefix of its witness sequence (that is, $W(\Sigma') = (Q_\pi(\Sigma), U)$). We will do this by applying swaps to *swappable pairs* in Σ.

▶ **Definition 28.** Consider a trajectory $\Sigma \in \mathcal{X}^\tau$. A pair $(\mathbf{w}, \mathbf{y})$ of named indices of flaws is called *a swappable* pair in Σ if it can be swapped in Σ (i.e., $W(\Sigma) = (\ldots \mathbf{w}, \mathbf{y} \ldots)$ and $\mathbf{w} \nsim \mathbf{y}$) and either

1. $(\mathbf{w}, \mathbf{y}) \in (\mathbf{V}(\Sigma) \setminus \mathbf{Q}(\Sigma)) \times \mathbf{Q}(\Sigma)$, or
2. $(\mathbf{w}, \mathbf{y}) \in \mathbf{Q}(\Sigma) \times \mathbf{Q}(\Sigma)$ and their order in $Q_\pi(\Sigma)$ is different: $Q_\pi(\Sigma) = (\ldots, \mathbf{y}, \mathbf{w}, \ldots)$

The position of the rightmost swappable pair in Σ will be denoted as $k(\Sigma)$, where the position of $(\mathbf{w}, \mathbf{y})$ in Σ is the number of named indices of flaws that precede $\mathbf{y}$ in $W(\Sigma)$. If Σ does not contain a swappable pair then $k(\Sigma) = 0$. Thus, $k(\Sigma) \in [0, |\Sigma| - 1]$.

We can only apply finite many swaps to swappable pairs in Σ. This is because swapping pairs of the first form moves a named index in $\mathbf{Q}(\Sigma)$ to the left, while swapping pairs of the second one decrease the number of pairs whose relative order in $Q(\Sigma)$ is not consistent with the one in $Q_\pi(\Sigma)$. Clearly, both of these actions can be performed only a finite number of times.

The following lemma shows how we can obtain a mapping Φ^τ such that $\mathcal{X}^\tau_\pi := \Phi^\tau(\mathcal{X}^\tau)$ satisfies the first condition of Lemma 26. The proof is identical (up to minor changes) to the one of Lemma 27 of [24].

▶ **Lemma 29.** *Consider a trajectory $\Sigma \in \mathcal{X}^\tau$ such that $W(\Sigma) = (A, U)$ where A and U are some sequences of indices of flaws, and there are no swappable pairs inside U. Then $U = (B, C)$ where sequence B is a subsequence of $Q_\pi(\Sigma)$ and C does not contain named indices of flaws from $\mathbf{Q}(\Sigma)$.*

In particular, if $|A| = 0$ and $W(\Sigma) = U$ does not contain a swappable pair then $W(\Sigma) = (Q_\pi(\Sigma), C)$.

It remains to show that Φ^τ can be constructed so that is also a bijection and that it satisfies the second condition of Lemma 26. To do so, consider the following algorithm.

- Let $\mathcal{X}_0 = \mathcal{X}^\tau$.
- While $k = \max_{\Sigma \in \mathcal{X}_p} k(\Sigma) \neq 0$
  - For each $\Sigma \in \mathcal{X}_p$: if $k(\Sigma) = k$ then swap the pair $(\mathbf{w}, \mathbf{y})$ at position k in Σ, otherwise leave Σ unchanged.
  - Let $\mathcal{X}_{p+1}$ the new set of trajectories.

For a witnesses sequence W define $\mathcal{X}_p[W] = \{\Sigma \in \mathcal{X}_p \mid Q_\pi(\Sigma) = W\}$ for an index $p \geq 0$. Now the following lemma concludes the proof since $\mathcal{X}_0[W] \subseteq \mathcal{X}^\tau$ is valid. Its proof is identical (up to minor changes) to the proof of Lemma 28 in [24].

▶ **Lemma 30.** *If set $\mathcal{X}_p[W]$ is valid then so is $\mathcal{X}_{p+1}[W]$, and the mapping from $\mathcal{X}_p[W]$ to $\mathcal{X}_{p+1}[W]$ defined by the algorithm above is injective.*

B Proof of Lemma 12

For the sake of brevity, we extend the notion of "addressing a flaw" to an arbitrary state $\sigma \in \Omega$, meaning that we recolor the vertices associated with the operation of addressing a flaw in the same way we would do it if the constraint corresponding to the flaw was indeed violated. Consider the following random experiments.

- Address B_v at an arbitrary state $\sigma \in \Omega$ to get a state σ'. Let $\Pr_\sigma[B_v]$ denote the probability that $\sigma' \in B_v$.
- Address Z_v at an arbitrary state $\sigma \in \Omega$ to get a state σ'. Let $\Pr_\sigma[Z_v]$ denote the probability that $\sigma' \in Z_v$.

Our claim now is that

$$\gamma(B_v) \leq \max_{\sigma' \in \Omega} \Pr_{\sigma'}[B_v] \ ; \tag{9}$$

$$\gamma(Z_v) \leq \max_{\sigma' \in \Omega} \Pr_{\sigma'}[Z_v] \ . \tag{10}$$

To see this, let $f_v \in \{B_v, Z_v\}$ and observe that

$$\gamma(f_v) = \max_{\sigma' \in \Omega} \sum_{\sigma \in f_v} \frac{\mu(\sigma)}{\mu(\sigma')} \rho_v(\sigma, \sigma') = \max_{\sigma' \in \Omega} \sum_{\sigma \in \mathrm{In}_{f_v}(\sigma')} \frac{1}{|\Lambda(\sigma)|} \ ,$$

where $\Lambda(\sigma) := \prod_{u \in N_v} L_u(\sigma)$ is the cartesian product of the lists of available colors of each vertex $u \in N_v$ at state σ and $\mathrm{In}_{f_v}(\sigma')$ is the set of states $\sigma \in f_v$ such that $\sigma' \in A(f_v, \sigma)$.

The key observation now is that $\Lambda(\sigma') = \Lambda(\sigma)$ for each state $\sigma \in \mathrm{In}_{f_v}(\sigma)$. This is because any transition of the form $\sigma \xrightarrow{f_v} \sigma'$ does not alter the lists of available colors of vertices $u \in N_v$, since the graph is triangle-free. Thus,

$$\gamma(f_v) = \max_{\sigma' \in \Omega} \frac{|\mathrm{In}_{f_v}(\sigma')|}{\Lambda(\sigma')} = \max_{\sigma' \in \Omega} \Pr[f_v] \ ,$$

where the second equality follows from the fact that there is a bijection between $\mathrm{In}_{f_v}(\sigma')$ and the set of color assignments from $\Lambda(\sigma')$ to the vertices of N_v that violate the constraint related to flaw f_v.

The following lemma concludes the proof.

▶ **Lemma 31** ([26]). *For every state $\sigma \in \Omega$ it holds that*

(a) $\Pr_\sigma[B_v] < \Delta^{-4}$;

(b) $\Pr_\sigma[Z_v] < \Delta^{-4}$.

The Cover Time of a Biased Random Walk on a Random Regular Graph of Odd Degree

Tony Johansson[1]
Department of Mathematics, Uppsala University, Uppsala, Sweden
tony.johansson@math.uu.se
https://orcid.org/0000-0002-9264-3462

Abstract

We consider a random walk process, introduced by Orenshtein and Shinkar [10], which prefers to visit previously unvisited edges, on the random r-regular graph G_r for any odd $r \geq 3$. We show that this random walk process has asymptotic vertex and edge cover times $\frac{1}{r-2}n\log n$ and $\frac{r}{2(r-2)}n\log n$, respectively, generalizing the result from [7] from $r=3$ to any larger odd r. This completes the study of the vertex cover time for fixed $r \geq 3$, with [3] having previously shown that G_r has vertex cover time asymptotic to $\frac{rn}{2}$ when $r \geq 4$ is even.

2012 ACM Subject Classification Theory of computation → Random walks and Markov chains

Keywords and phrases Random walk, random regular graph, cover time

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.45

1 Introduction

We consider a biased random walk on the random r-regular n-vertex graph G_r for any odd fixed $r \geq 5$, i.e. a graph chosen uniformly at random from the set of r-regular graph on an even number n of vertices. In short, this is a random walk which chooses a previously unvisited edge whenever possible, see Section 2 for a precise definition. This process was introduced by Orenshtein and Shinkar [10]. In [7] it is shown that with high probability, G_3 is such that the expected vertex cover time $C_V^b(G_3)$ and expected edge cover time $C_E^b(G_3)$ of the biased random walk satisfy[2]

$$C_V^b(G_3) \sim n\log n, \quad C_E^b(G_3) \sim \frac{3}{2}n\log n.$$

We generalize this result as follows.

▶ **Theorem 1.** *Suppose $r \geq 3$ is odd, and let G_r be chosen uniformly at random from the set of r-regular graphs on n vertices. Then with high probability, G_r is such that*

$$C_V^b(G_r) \sim \frac{1}{r-2}n\log n, \quad C_E^b(G_r) \sim \frac{r}{2(r-2)}n\log n.$$

With this the asymptotic leading term of $C_V^b(G_r)$ is known for all $r \geq 3$, with Berenbrink, Cooper and Friedetzky [3] having previously shown that $C_V^b(G_r) \sim \frac{rn}{2}$ for any even $r \geq 4$. They also showed that for even r, $C_E^b(G_r) = O(\omega n)$ for any ω tending to infinity with n, with the ω factor owing to the w.h.p.[3] existence of cycles of length up to ω.

[1] Supported in part by the Knut and Alice Wallenberg Foundation.
[2] We say that $a_n \sim b_n$ if $\lim a_n/b_n = 1$.
[3] An event $\mathcal{E}$ holds *with high probability* (w.h.p.) if $\Pr\{\mathcal{E}\} \to 0$ as $n \to \infty$.

Cooper and Frieze [5] considered the simple random walk on G_r, showing that for any $r \geq 3$, $C_V^s(G_r) \sim \frac{r-1}{r-2} n \log n$ and $C_E^s(G_r) \sim \frac{r(r-1)}{2(r-2)} n \log n$, and we see that the biased random walk speeds up the cover time by a factor of $1/(r-1)$ for odd r. Cooper and Frieze [6] also consider the non-backtracking random walk, i.e. the walk which at no point reuses the edge used in the previous step, showing that $C_V^{nb}(G_r) \sim n \log n$ and $C_E^{nb}(G_r) \sim \frac{r}{2} n \log n$. Here, the biased random walk gains a factor of $1/(r-2)$ for odd r.

Theorem 1 will follow from the following theorem. For a graph G let $C_V^b(G;s)$ ($C_E^b(G;t)$) denote the expected time taken for the biased random walk to visit s vertices (t edges) of G. Note that $C_\cdot^b(G;\cdot)$ is defined as an expectation over the space of random walks on the fixed graph G, and that $\mathbb{E}\left(C_\cdot^b(G_r;\cdot)\right)$ takes the expectation of $C_\cdot^b(G;\cdot)$ when G is chosen uniformly at random from the set of r-regular graphs.

▶ **Theorem 2.** *Suppose $r \geq 3$ is odd, and suppose G_r is chosen uniformly at random from the set of r-regular graphs on an even number n of vertices. Let $n - n\log^{-1/2} n \leq s \leq n$ and $(1-\log^{-1/2} n)\frac{rn}{2} \leq t \leq rn/2$, and let $\varepsilon > 0$. Then*

$$\mathbb{E}\left(C_V^b(G_r;s)\right) = \frac{1 \pm \varepsilon}{r-2} n \log\left(\frac{n}{n-s+1}\right) + o(n\log n),$$
$$\mathbb{E}\left(C_E^b(G_r;t)\right) = \frac{r \pm \varepsilon}{2(r-2)} n \log\left(\frac{rn}{rn-2t+1}\right) + o(n\log n).$$

We take $a = b \pm c$ to mean that $b - c < a < b + c$. The $(1 - \log^{-1/2} n)$ factor in the lower bounds for s, t is a fairly arbitrary choice, and the proof here is valid for any $(1 - 1/\omega)$ factor with ω tending to infinity sufficiently slowly. The specific choice of $\log^{-1/2} n$ is made to aid readability.

Applying Theorem 2 with $s = n$ and $t = rn/2$ gives $\mathbb{E}\left(C_V^b(G_r)\right) \sim \frac{1}{r-2} n \log n$ and $\mathbb{E}\left(C_E^b(G_r)\right) \sim \frac{r}{2(r-2)} n \log n$. A little extra work is needed to conclude that w.h.p. G_r is such that $C_V^b(G_r), C_E^b(G_r)$ have the same asymptotic values. We refer to the full paper version of [7], where this is done in detail.

2 Proof outline

The random r-regular graph G_r is chosen according to the *configuration model*, introduced by Bollobás [4]. Each vertex $v \in [n]$ is associated with a set $\mathcal{P}(v)$ of r *configuration points*, and we let $\mathcal{P} = \cup_v \mathcal{P}(v)$. We choose u.a.r. (*uniformly at random*) a perfect matching μ of the points in $\mathcal{P}$. Each μ induces a multigraph G on $[n]$ in which u is adjacent to v if and only if $\mu(x) \in \mathcal{P}(v)$ for some $x \in \mathcal{P}(u)$, allowing parallel edges and self-loops. Any simple r-regular graph is equally likely to be chosen under this model.

We study a *biased random walk*. On a fixed graph G, this process is defined as follows. Initially, all edges are declarded *unvisited*, and we choose a vertex v_0 uniformly at random as the *active* vertex. At any point of the walk, the walk moves from the active vertex v along an edge chosen uniformly at random from the unvisited edges incident to v, after which the edge is permanently declared *visited*. If there are no unvisited edges incident to v, the walk moves along a visited edge chosen uniformly at random. The other endpoint of the chosen edge is declared active, and the process is repeated.

A biased random walk on the random r-regular graph can be seen as a random walk on the configuration model, where we expose μ along with the walk as follows. Initially choosing some point $x_0 \in \mathcal{P}$ u.a.r., we walk to $x_1 = \mu(x_0)$, chosen u.a.r. from $\mathcal{P} \setminus \{x_1\}$. Suppose $x_1 \in \mathcal{P}(v_1)$. From x_1 the walk moves to some unvisited $x_2 \in \mathcal{P}(v_1)$. In general, if

$W_k = (x_0, x_1, \ldots, x_k)$ then (i) if k is odd, the walk moves to $x_{k+1} = \mu(x_k)$ (chosen u.a.r. from $\mathcal{P} \setminus \{x_0, \ldots, x_k\}$ if x_k is previously unvisited), and (ii) if k is even, the walk moves from $x_k \in \mathcal{P}(v_k)$ to $x_{k+1} \in \mathcal{P}(v_k)$, chosen u.a.r. from the unvisited points of $\mathcal{P}(v_k)$ if such exist, otherwise chosen u.a.r. from all of $\mathcal{P}(v_k)$.

We define $C(t)$ to be the number of steps taken immediately before the walk exposes its tth distinct edge. To be precise, if $W_k = (x_0, \ldots, x_k)$ denotes the walk after k steps, then

$$C(t) = \min\{k : |\{x_0, x_1, \ldots, x_k\}| = 2t - 1\}.$$

Note that this set consists of exactly one k, as the walk will immediately go to $x_{C(t)+1} = \mu(x_k)$, which has not been visited before. We also let $W(t) = W_{C(t)}$. Note that $C(t)$ is a random variable over the combined probability space of random graphs and random walks, as opposed to $C_V^b(G_r)$ and $C_E^b(G_r)$ which are variables over the space of random graphs only. We will show (Lemma 8) that if $t_1 = (1 - \log^{-1/2} n)\frac{rn}{2}$ then

$$\mathbb{E}\,(C(t_1)) = o(n \log n),$$

which does not contribute significantly to the cover time. The main part of the proof is calculating $\mathbb{E}\,(C(t+1) - C(t))$ when $t \geq t_1$. We define the random graph $G(t) \subseteq G_r$ as the graph spanned by the first t distinct edges visited by the walk. If, immediately after discovering its tth edge, the biased random walk inhabits a vertex incident to no unvisited edges, then a simple random walk commences on $G(t)$, and $C(t+1) - C(t)$ is the number of steps taken for this random walk to hit a vertex incident to an unvisited edge.

We construct from $G(t)$ a graph $G^*(t)$ by contracting all vertices incident to at least one unvisited edge into one "supervertex" x. Thus, conditioning on $W(t)$, the graph $G^*(t)$ is a fixed graph, i.e. one with no random edges. We will show that when $t \geq t_1$, w.h.p. x lies on "few" cycles of "short" length and has the appropriate number of self-loops (to be made precise in Section 4), which will imply that the expected hitting time of x for a simple random walk on $G^*(t)$ is

$$\mathbb{E}\,(H(x)) \sim \frac{1}{r-2}\frac{rn}{rn-2t}.$$

The paper is laid out as follows. Sections 3, 4 and 5 respectively discuss properties of the random regular graph, hitting times of simple random walks, and a uniformity lemma for biased random walks, and may be read in any order. Section 6 contains the calculation of the cover time. Appendix A and B are devoted to bounding the sizes of certain sets appearing in the calculations.

3 Properties of G_r

Here we collect some properties of random r-regular graphs, chosen according to the configuration model.

▶ **Lemma 3.** *Let $r \geq 3$. Let G_r denote the random r-regular graph on vertex set $[n]$, chosen according to the configuration model. Let ω tend to infinity arbitrarily slowly with n. Its value will always be small enough so that where necessary, it is dominated by other quantities that also go to infinity with n.*

- **(i)** *With high probability, the second largest in absolute value of the eigenvalues of the transition matrix for a simple random walk on G_r is at most 0.99.*
- **(ii)** *With high probability, G_r contains at most ωr^ω cycles of length at most ω,*
- **(iii)** *The probability that G_r is simple is $\Omega(1)$.*

Friedman [8] showed that for any $\varepsilon > 0$, the second eigenvalue of the transition matrix is at most $2\sqrt{r-1}/r + \varepsilon$ w.h.p., which gives (i). Property (ii) follows from the Markov inequality, given that the expected number of cycles of length $k \leq \omega$ can be bounded by $O(r^k)$. For the proof of (iii) see Frieze and Karoński [9], Theorem 10.3. Note that (iii) implies that any property which holds w.h.p. for the configuration multigraph holds w.h.p. for simple r-regular graphs chosen uniformly at random.

Let $G(t)$ denote the random graph formed by the edges visited by $W(t)$. Let $X_i(t)$ denote the set of vertices incident to i unvisited edges in $G(t)$ for $i = 0, 1, \ldots, r$. Let $\overline{X}(t) = X_1(t) \cup \cdots \cup \cup X_r(t)$ denote the set of vertices incident to at least one unvisited edge. Let $G^*(t)$ denote the graph obtained from $G(t)$ by contracting the set $\overline{X}(t)$ into a single vertex, retaining all edges. Define $\lambda^*(t)$ to be the second largest eigenvalue of the transition matrix for a simple random walk on $G^*(t)$.

We note that by [2, Corollary 3.27], if Γ is a graph obtained from G by contracting a set of vertices, retaining all edges, then $\lambda(\Gamma) \leq \lambda(G)$. This implies that $\lambda^*(t) = \lambda(G^*(t)) \leq \lambda(G) \leq 0.99$ for all t. Initially, for small t, we find that w.h.p. $G^*(t)$ consists of a single vertex. In this case there is no second eigenvalue and we take $\lambda^*(t) = 0$. This is in line with the fact that a random walk on a one vertex graph is always in the steady state.

We define $C(t)$ to be the number of steps the biased random walk takes to traverse t distinct edges of G_r. Of course, if G_r is disconnected and the random walk starts in a connected component of less than t edges, then $C(t) = \infty$. We resolve this by defining a stopping time $T^* = \min\{t : \lambda^*(t) > 0.99\}$, and setting $C^*(t) = C(\min\{t, T\})$. Strictly speaking, the estimates of $C(t)$ in the upcoming sections are estimates of $C^*(t)$, but we do not make any explicit distinction between the two, noting that by Lemma 3 (i), w.h.p. $T^* = \infty$ which implies that $C^*(t) = C(t)$ for all t.

4 Hitting times in simple random walks

We are interested in calculating $C(t+1) - C(t)$, i.e. the time taken between discovering the tth and the $(t+1)$th edge. Between the two discoveries, the biased random walk can be coupled to a simple random walk on the graph induced by $W(t)$, and in this section we derive the hitting time of a certain type of expanding vertex set.

▶ **Definition 4.** Let $G = (V, E)$ be an r-regular graph. A set $S \subseteq V$ is a *root set of order* ℓ if (i) $|S| \geq \ell^5$, (ii) the number of edges with both endpoints in S is between $|S|/2$ and $(1/2 + \ell^{-3})|S|$, and (iii) there are at most $|S|/\ell^3$ paths of length at most ℓ between vertices of S which use no edges fully contained in S.

The following lemma establishes the hitting time of root sets.

▶ **Lemma 5.** *Let ω tend to infinity arbitrarily slowly with n. Suppose G is an r-regular graph on n vertices whose transition matrix has second largest eigenvalue $\lambda \leq 0.99$, containing at most ωr^ω cycles of length at most ω. If S is a root set of order ω and a simple random walk is initiated at a uniformly random vertex of G, then the expected number of steps needed to reach S is*

$$\mathbb{E}\,(H(S)) \sim \frac{r}{r-2}\frac{n}{|S|}.$$

The full proof of Lemma 5 is omitted in this extended abstract. The proof is based on the following (see e.g. [2, Lemma 2.11]). If $|S| = n/\omega$ for some ω tending to infinity with n, then

$$\mathbb{E}\,(H(S)) = \frac{n}{|S|} Z_{SS}.$$

Here Z_{SS} is a constant which can be approximated by the expected number of times a walk starting in S visits S in its first ω steps. We show that this expectation is approximately $r/(r-2)$ for root sets of order ω.

The following lemma is an important step in generalizing Theorem 1 from $r = 3$ to larger r. It follows from reversibility properties of random walks on regular graphs, and the proof is omitted in this extended abstract.

▶ **Lemma 6.** *Let G be an r-regular graph with positive eigenvalue gap. Let $R \subseteq S \subseteq V$ be vertex sets. Suppose a simple random walk is initiated at a uniformly random vertex $y \in R$, and ends as soon as it hits $S \setminus \{y\}$. Then there is a constant $B > 0$ such that for any $x \in S$, the probability that the walk ends at x is at most $B/|R|$.*

5 The structure of $\overline{X}$

The walk $W(t)$ induces a colouring on the edges and vertices of G_r as follows. An edge is coloured red, green or blue if it has been visited zero, one or at least two time(s), respectively. A vertex is (i) green if it is incident to exactly $r-1$ green edges and one red edge, (ii) red if it is incident to red edges only, and (iii) blue otherwise.

Recall that $X_i(t)$ denotes the set of vertices incident to exactly i red edges in $W(t)$. We let $X_1^g(t)$, $X_1^b(t)$ denote the green and blue vertices of $X_1(t)$, respectively, and set

$$Z(t) = X_1^b(t) \cup \bigcup_{i=2}^{r} X_i(t).$$

The green edges and vertices are of particular interest. Suppose $e_1 = (u, v), e_2 = (v, w)$ are consecutive green edges in the walk $W(t)$, meeting at a vertex v. Let $p, q \in \mathcal{P}(v)$ denote the configuration points in v of e_1, e_2, respectively. We call the pair (p, q) a *green link* if v is a green vertex.

Given a walk $W(t)$, we form the *contracted walk* $\langle W(t) \rangle$ as follows. For any green link (p, q), replace the corresponding edges $(u, v), (v, w)$ by the edge (u, w) (coloured green), freeing the configuration points p, q. This is repeated until there are no green links left. Note that if $e_1, e_2, \ldots, e_k$ are green edges visited sequentially by the walk where e_i, e_{i+1} share a green link, then at the end of the process the entire path is replaced by one green edge.

Let $L(W)$ denote the set of green links in the walk W, so $L(W) \subseteq \mathcal{P} \times \mathcal{P}$ is a set of ordered pairs of configuration points. Say that two walks W_1, W_2 are equivalent if $\langle W_1 \rangle = \langle W_2 \rangle$ and $L(W_1) = L(W_2)$. The equivalence class is denoted $[W] = (\langle W \rangle, L(W))$. The next lemma shows that equivalent walks are equiprobable.

▶ **Lemma 7.** *If W is such that $Pr\{[W(t)] = [W]\} > 0$, then*

$$Pr\{W(t) = W \mid [W(t)] = [W]\} = \frac{1}{|[W]|}.$$

Proof. Let W be a walk with $\Pr\{W(t) = W\} > 0$. We can calculate the probability of $W(t) = W$ exactly. There are two different types of steps a walk can take. Suppose the walk has visited t distinct edges.

- If the walk occupies a vertex incident to no red edges, it chooses an edge with probability r^{-1}.
- If the walk occupies a vertex incident to k red edges, it chooses one of the k red edges with probability k^{-1}. The other endpoint of the red edge is chosen uniformly at random from $rn - 2t - 1$ configuration points.

The probability of $W(t) = W$ is

$$\Pr\{W(t) = W\} = \frac{1}{rn} \prod_{k=2}^{r} k^{-i_k} \prod_{s=0}^{t} \frac{1}{rn - 2s - 1},$$

for some integers $i_2, \ldots, i_r \geq 0$, counting the number of steps of the different types. The $1/rn$ factor accounts for the starting point of the walk. Now, if $W_1 \sim W_2$, then W_1 and W_2 contain the same number of edges, and $i_k(W_1) = i_k(W_2)$ for $k = 2, \ldots, r$. Indeed, W_1 and W_2 only disagree in which order they visit the links in L. ◀

We can now view the biased random walk as a walk on the equivalence class $[W(t)]$. Any time a green edge in $[W(t)]$ is visited, the probability that the edge corresponds to a green link in a randomly chosen $W(t) \in [W(t)]$ is about $L(t)/\Phi(t)$, where $L(t)$ is the number of green links in $[W(t)]$ and $\Phi(t)$ the number of green edges in $W(t)$. This along with bounds for $X_1^g(t)$ and $Z(t)$ provides a precise recursion for $\mathbb{E}(\Phi(t))$, which we use to prove the following. W.h.p., if $t = (1-\delta)\frac{rn}{2}$,

$$|X_1^g(t)| \sim rn\delta \quad \text{when } \delta \leq \log^{-1/2} n, \tag{1}$$

$$|Z(t)| = O(n\delta^{3/2}) \quad \text{when } \delta \leq \log^{-1/2} n, \tag{2}$$

$$\Phi(t) \geq n\delta^{1-\alpha} \quad \text{when } n^{-4/5+\beta} \leq \delta \leq \log^{-1/2} n, \tag{3}$$

where $a > 0$ and $0 < \beta < 1/20$ are constants. Note in particular that $Z(t) \ll X_1^g(t) \ll \Phi(t)$ in the ranges where these bounds apply. Details are found in Appendix A and B.

Suppose $n^{-4/5+\beta} \leq \delta \leq \log^{-1/2} n$. As $L(t) = \frac{r-1}{2} X_1^g(t) = o(\Phi(t))$, when $W(t) \in [W(t)]$ is chosen uniformly at random, the links of $L(t)$ are sprinkled into the much larger set of green edges, and are expected to be spread far apart. This will imply that $X_1^g(t)$ is a root set of order ω, and as $X_1^g(t)$ makes up almost all of $\overline{X}(t)$ by (1) and (2), the set $\overline{X}(t)$ is also a root set of order ω. When $\delta \leq n^{-4/5+\beta}$, the same technique can be applied with a little more work.

6 Calculating the cover time

Define

$$\delta_0 = \frac{1}{\log\log n},\ \delta_1 = \frac{1}{\log^{1/2} n},\ \delta_2 = \frac{1}{\log^2 n},\ \delta_3 = n^{-3/4},\ \delta_4 = n^{-1}\log n, \tag{4}$$

and $t_i = (1-\delta_i)\frac{rn}{2}$ for $i = 0, 1, 2, 3$. From this point on we will use t and δ interchangeably to denote time, and the two are always related by $t = (1-\delta)\frac{rn}{2}$. We begin by showing that the time taken to find the first t_1 edges contributes insignificantly to the cover time.

▶ **Lemma 8.**

$$\mathbb{E}(C(t_1)) = o(n \log n).$$

This is proved in Section 6.1. We then move on to estimating the expected cover time increment for larger t.

▶ **Lemma 9.** *For $t_1 \leq t \leq t_4$ and any $\varepsilon > 0$,*

$$\mathbb{E}(C(t+1) - C(t)) = \left(\frac{r}{r-2} \pm \varepsilon\right) \frac{n}{rn - 2t}.$$

The time to discover the final $O(\log n)$ edges can be bounded as follows:

$$\mathbb{E}\left(C\left(\frac{rn}{2}\right) - C(t_4)\right) \leq \sum_{t=t_4}^{rn/2-1} O\left(\frac{n}{rn-2s}\right) = o(n\log n).$$

The proof of Lemma 9 is based on the following calculation. Define events

$$\mathcal{A}(t) = \left\{|X_1^g(t) - (rn-2t)| \leq \frac{rn-2t}{\omega}\right\},$$
$$\mathcal{B}(t) = \{\overline{X}(t) \text{ is a root set of order } \omega\},$$

and set $\mathcal{E}(t) = \mathcal{A}(t) \cap \mathcal{B}(t)$. Then for any $\varepsilon > 0$, $\mathbb{E}\left(C(t+1) - C(t)\right)$ can be calculated as

$$\left(\frac{r}{r-2} \pm \varepsilon\right)\frac{n}{rn-2t}\Pr\left\{\mathcal{E}(t)\right\} + O\left(\frac{n}{rn-2t}\Pr\left\{\overline{\mathcal{E}(t)}\right\}\right) + O(\log n).$$

Indeed, suppose $\mathcal{E}(t)$ holds. As $X_1(t)$ contains almost all unvisited configuration points, edge t is attached to some $v \in X_1(t)$ w.h.p., and a simple random walk commences at v, ending once it hits $\overline{X} \setminus \{v\}$. As the vertices of $\overline{X}$ are spread far apart, it is unlikely that this happens within $O(\log n)$ steps. After a logarithmic number of steps, the random walk has mixed to within ε of the stationary distribution π in total variation. Lemma 5 shows that after this point, the expected time taken to hit $\overline{X}$ is $(r/(r-2) \pm \varepsilon)n/|\overline{X}|$, and as $\mathcal{A}(t)$ holds we have $|\overline{X}| \sim (rn-2t)$. If $\mathcal{E}(t)$ does not hold, then we use the fact that the hitting time in a regular graph with positive eigenvalue gap is $O(n/|\overline{X}|) = O(n/(rn-2t))$ (as $|\overline{X}| \geq (rn-2t)/r$) as long as the graph has a positive eigenvalue gap. We refer to the discussion in Section 3 justifying our assumption that the second largest eigenvalue stays at most 0.99 throughout the process. Lemma 9 will now follow from proving that $\Pr\{\mathcal{E}(t)\} = 1 - o(1)$ for any fixed $t_1 \leq t \leq t_4$. This is done in Section 6.2.

6.1 Phase one: Proof of Lemma 8

With t_1 as in (4), we show that $\mathbb{E}\left(C(t_1)\right) = o(n\log n)$. Suppose $W(t) = (x_0, x_2, \ldots, x_k)$ for some t, k. If $x_k \in \mathcal{P}(\overline{X}(t))$ then $x_{k+1} = \mu(x_k)$ is uniformly random inside $\mathcal{P}(\overline{X}(t)) \setminus \{x_k\}$, and since $C(t+1) = C(t) + 1$ in the event of $x_{k+1} \in \mathcal{P}(X_2 \cup \cdots \cup X_r)$, we have

$$\mathbb{E}\left(C(t+1) - C(t)\right) \leq 1 + \mathbb{E}\left(C(t+1) - C(t) \mid x_{k+1} \in \mathcal{P}(X_1)\right)\Pr\left\{x_{k+1} \in \mathcal{P}(X_1)\right\}, \quad (5)$$

We use the following theorem of Ajtai, Komlós and Szemerédi [1] to bound the expected change when $x_{k+1} \in \mathcal{P}(X_1)$.

▶ **Theorem 10.** *Let $G = (V, E)$ be an r-regular graph on n vertices, and suppose that each of the eigenvalues of the adjacency matrix with the exception of the first eigenvalue are at most λ_G (in absolute value). Let A be a set of cn vertices of G. Then for every ℓ, the number of walks of length ℓ in G which avoid A does not exceed $(1-c)n((1-c)r + c\lambda_G)^\ell$.*

The set A of Theorem 10 is fixed. In our case we choose a point x_{k+1} uniformly at random from $\mathcal{P}(X_1(t))$, so we consider a simple random walk initiated at a uniformly random vertex $u \in X_1(t)$. The subsequent walk now begins at vertex u and continues until it hits a vertex of $Y_u = \overline{X}(t) \setminus \{u\}$. Because the vertex u is random, the set Y_u differs for each possible exit vertex $u \in X_1(t)$. To apply Theorem 10, we split $X_1(t)$ into two disjoint sets A, A' of (almost) equal size. For $u \in A$, instead of considering the number of steps needed to hit Y_u, we can upper bound this by the number of steps needed to hit $B' = A' \cup X_2 \cup \cdots \cup X_r$, and vice versa. Suppose without loss of generality that $u \in A$.

Let $S(\ell)$ be a simple random walk of length ℓ starting from a uniformly chosen vertex of A. Thus $S(\ell)$ could be any of $|A|r^\ell$ uniformly chosen random walks. Let $c = |B'|/n$. The probability p_ℓ that a randomly chosen walk of length ℓ starting from A has avoided B' is, by Theorem 10, at most

$$p_\ell \le \frac{1}{(|X_1(t)|/2)r^\ell}(1-c)n(r(1-c)+c\lambda_G)^\ell \le \frac{2(1-c)n}{|X_1(t)|}((1-c)+c\lambda)^\ell,$$

where $\lambda \le .99$ (see Lemma 3) is the absolute value of the second largest eigenvalue of the transition matrix of S. Thus

$$\mathbb{E}_A\left(H(C)\right) \le \sum_{\ell\ge 1} p_\ell \le \frac{2(1-c)n}{|X_1(t)|}\frac{1}{c(1-\lambda)}. \tag{6}$$

So,

$$\mathbb{E}\left(C(t+1)-C(t) \mid x_{2k} \in \mathcal{P}(X_1(t))\right) = O\left(\frac{(n-|B'|)n}{|X_1||B'|}\right). \tag{7}$$

Now, for any t we have $r^{-1}(rn-2t) \le |B'| \le rn-2t$, so summing over $0 \le t \le t_1$, (5) gives $\mathbb{E}\left(C(t_1)\right) = o(n\log n)$.

6.2 Phase two: Proof of Lemma 9, $t_1 \le t < t_3$

Let ω tend to infinity arbitrarily slowly with n and define for $t \ge t_1$,

$$\mathcal{A}(t) = \left\{|X_1^g(t) - (rn-2t)| \le \frac{rn-2t}{\omega}\right\},$$
$$\mathcal{B}(t) = \{\overline{X}(t) \text{ is a root set of order } \omega\},$$

and set $\mathcal{E}(t) = \mathcal{A}(t) \cap \mathcal{B}(t)$. As discussed above, it remains to prove the following lemma.

▶ **Lemma 11.** *Fix $t_1 \le t \le t_4$. Then*

$$Pr\{\mathcal{E}(t)\} = 1 - o(1).$$

Proof. First fix $t_1 \le t \le t_3$. By (1) – (3), for some $\alpha > 0$, the following holds w.h.p.:

$$\Phi(t) \ge n\delta^{1-\alpha},$$
$$X_1^g(t) = rn\delta(1-O(\delta^{1/2})),$$
$$Z(t) = O(n\delta^{3/2}).$$

Condition on some $[W(t)]$ satisfying these values. We will distribute the links $L(t)$ into the green edges to form $W(t)$. Suppose $\ell_1 \in L(t)$ is placed at some green edge e_1. As there are at most $Z(t)r^\omega$ green edges within distance ω of $Z(t)$, the probability that it is placed within distance ω of $Z(t)$ is $O(Z(t)r^\omega/\Phi(t)) = o(1)$. The probability that any particular $\ell_2 \in L(t)$ is placed on one of the $O(r^\omega)$ green edges within distance ω of e_1 is $O(r^\omega/\Phi(t))$. Let $D(\ell_1,\ell_2)$ be the distance in $[W(t)]$ between ℓ_1 and ℓ_2. Then

$$\sum_{\ell_1\ne\ell_2} \Pr\left\{D(\ell_1,\ell_2) \le \omega\right\} = O\left(\frac{|L(t)|^2 r^\omega}{\Phi(t)}\right) = O\left(n\delta^{2-\frac{1+\varepsilon}{r-1}}3^\omega\right) = o(n\delta).$$

This shows that all but $o(n\delta)$ vertices in $\overline{X}(t)$ are $v \in X_1^g(t)$ with $d(v,\overline{X}(t)) > \omega$. By Lemma 3, at most $\omega r^\omega = o(n\delta)$ vertices in G lie on cycles of length at most ω. This shows that w.h.p., $\overline{X}(t)$ is a root set of order ω.

For $t_3 \le t \le t_4$ we can no longer use the bound (3) for $\Phi(t)$, but instead we can show that w.h.p., the conditions of $\mathcal{E}(t_3)$ hold with enough room to spare that they must hold also for t. For example, $Z(t_3)$ is empty w.h.p., so $Z(t) \subseteq Z(t_3)$ must also be empty. ◀

References

1. Miklós Ajtai, János Komlós, and Endre Szemerédi. Deterministic simulation in LOGSPACE. In Alfred V. Aho, editor, *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 132–140. ACM, 1987. doi:10.1145/28395.28410.
2. David Aldous and James Allen Fill. Reversible markov chains and random walks on graphs, 2002. Unfinished monograph, recompiled 2014, available at http://www.stat.berkeley.edu/~aldous/RWG/book.html.
3. Petra Berenbrink, Colin Cooper, and Tom Friedetzky. Random walks which prefer unvisited edges: Exploring high girth even degree expanders in linear time. *Random Struct. Algorithms*, 46(1):36–54, 2015. doi:10.1002/rsa.20504.
4. Béla Bollobás. A probabilistic proof of an asymptotic formula for the number of labelled regular graphs. *Eur. J. Comb.*, 1(4):311–316, 1980. doi:10.1016/S0195-6698(80)80030-8.
5. Colin Cooper and Alan M. Frieze. The cover time of random regular graphs. *SIAM J. Discrete Math.*, 18(4):728–740, 2005. doi:10.1137/S0895480103428478.
6. Colin Cooper and Alan M. Frieze. Vacant sets and vacant nets: Component structures induced by a random walk. *SIAM J. Discrete Math.*, 30(1):166–205, 2016. doi:10.1137/14097937X.
7. Colin Cooper, Alan M. Frieze, and Tony Johansson. The cover time of a biased random walk on a random cubic graph. To appear in Proceedings of AofA 2018, preprint available at https://arxiv.org/abs/1801.00760.
8. Joel Friedman. A proof of Alon's second eigenvalue conjecture. In *Proceedings of the Thirty-fifth Annual ACM Symposium on Theory of Computing*, STOC '03, pages 720–724, New York, NY, USA, 2003. ACM. doi:10.1145/780542.780646.
9. Alan M. Frieze and Michal Karonski. *Introduction to Random Graphs*. Cambridge University Press, Cambridge, UK, 2015. doi:10.1017/CBO9781316339831.011.
10. Tal Orenshtein and Igor Shinkar. Greedy random walk. *Combinatorics, Probability & Computing*, 23(2):269–289, 2014. doi:10.1017/S0963548313000552.

A Set sizes

Recall the definition

$$Z(t) = X_1^b(t) \cup \bigcup_{i=2}^{r} X_i(t),$$

where X_i denotes the set of vertices incident to i unvisited edges, and X_1^b is the set of vertices in X_1 which are incident to at least one edge which has been visited more than once.

▶ **Lemma 12.** *There exists a constant $B > 0$ such that for $t \geq t_0$ and $0 < \theta = o(1)$,*

$$\mathbb{E}\left(e^{\theta Z(t)}\right) \leq \exp\left\{\theta B n \delta^{3/2}\right\}.$$

Proof. We show that there exists a $B > 0$ such that for any $m \geq 1$,

$$\Pr\{[m] \subseteq Z(t)\} \leq (B\delta)^{3m/2},$$

beginning with $m = 1$ before the general statement. Let $\mathcal{L} = \mathcal{L}(r)$ denote the set of vectors $(\ell_1, \ell_2, \ldots, \ell_k)$ with $\ell_i \in \{1, 2\}$ such that $\sum \ell_i \leq r - 1$, including in $\mathcal{L}$ the empty vector $\emptyset$,

excluding the vector $(2, 2, \ldots, 2)$ consisting of $(r-1)/2$ copies of 2 (which corresponds to X_1^g, as we will see). We partition

$$Z(t) = \bigcup_{\ell \in \mathcal{L}} Z_\ell(t),$$

where $v \in Z_\ell(t)$ for $\ell = (\ell_1, \ldots, \ell_k)$ if and only if there exists a sequence $0 < s_1 < s_2 < \cdots < s_k \le t$ such that v moves from $X_{r-\ell_1-\cdots-\ell_{j-1}}$ to $X_{r-\ell_1-\cdots-\ell_j}$ at time s_j for $j = 1, \ldots, k$, and is in $X_{r-\ell_1-\cdots-\ell_k}$ at time t. If $v \in X_i$ at time s, the probability that v is chosen by random assignment is $i/(rn-2s)$, while Lemma 6 shows that the probability that v is at the end of a blue walk is $O(1/(rn-2s))$. In either case, the probability that v moves from one set to another is at most $B/(rn-2s)$ for some $B > 0$. For a fixed $\ell = (\ell_1, \ldots, \ell_k) \in \mathcal{L}$, with $s_0 = 1$,

$$\Pr\{1 \in Z_\ell(t)\} \le \sum_{s_1 < \cdots < s_k} \prod_{j=1}^{k} \left[\prod_{s=s_{j-1}+1}^{s_j - 1} \left(1 - \frac{r - (\ell_1 + \cdots + \ell_{j-1})}{rn - 2s}\right) \frac{B}{rn - 2s_j} \right] \times \prod_{s=s_k+1}^{t} \left(1 - \frac{r - (\ell_1 + \cdots + \ell_k)}{rn - 2s}\right). \tag{8}$$

For $b \ge 1$ we use the bound

$$\prod_{s=t_0}^{t} \left(1 - \frac{b}{rn - 2s}\right) \le \left(\frac{rn - 2t}{rn - 2t_0}\right)^{b/2}. \tag{9}$$

Combining (8) and (9), the probability that $1 \in Z_\ell(t)$ is bounded above by

$$\sum_{s_1 < \cdots < s_k} \left[\prod_{j=1}^{k} \frac{B}{rn - 2s_j} \left(\frac{rn - 2s_j}{rn - 2s_{j-1}}\right)^{(r-(\ell_1+\cdots+\ell_{j-1}))/2} \right] \left(\frac{rn - 2t}{rn - 2s_k}\right)^{(r-(\ell_1+\cdots+\ell_k))/2}. \tag{10}$$

Collecting powers of $rn - 2s_j$ for $j = 1, \ldots, k$, we have

$$\Pr\{1 \in Z_\ell(t)\} \le B^k \frac{(rn-2t)^{(r-(\ell_1+\cdots+\ell_k))/2}}{(rn)^{r/2}} \sum_{s_1 < \cdots < s_k} \prod_{j=1}^{k} (rn - 2s_j)^{\ell_j/2-1}.$$

Let N denote the number of indices $j \in \{1, \ldots, k\}$ with $\ell_j = 1$. Then

$$\sum_{s_1 < \cdots < s_k} \prod_{j=1}^{k} (rn - 2s_j)^{\ell_j/2-1} \le \prod_{j=1}^{k} \left(\sum_{s=0}^{t} (rn - 2s_j)^{\ell_j/2-1} \right) \le n^{k-N} (rn - 2t)^{N/2},$$

which implies that

$$\Pr\{1 \in Z_\ell(t)\} \le \frac{B^k}{r^{r/2}} (rn-2t)^{(r+N-(\ell_1+\cdots+\ell_k))/2} n^{k-N-r/2}.$$

As $\ell_1 + \cdots + \ell_k = 2k - N$, we have $(r + N - (\ell_1 + \cdots + \ell_k))/2 = r/2 - k + N$. So

$$\Pr\{1 \in Z_\ell(t)\} \le \frac{B^k}{r^{k-N}} \delta^{r/2-k+N}.$$

We now argue that $r/2 - k + N \ge 3/2$, or equivalently $2(k-N) \le r-3$, for all $\ell \in \mathcal{L}$. Firstly, if $\ell_1 + \cdots + \ell_k \le r-3$ then we have $2(k-N) \le 2k - N = \ell_1 + \cdots + \ell_k \le r-3$. Secondly, if

$\ell_1+\cdots+\ell_k = r-2$ then as $r-2$ is odd we have $N \geq 1$, so $2(k-N) \leq 2k-N-1 \leq r-3$. Finally, if $\ell_1+\cdots+\ell_k = r-1$ then (as $(2,2,\ldots,2) \notin \mathcal{L}$) we have $N \geq 2$, so $2(k-N) \leq 2k-N-2 \leq r-3$.

As $|\mathcal{L}(r)|$ is a function of r only, and therefore constant with respect to n, it follows that

$$\Pr\{1 \in Z(t)\} = \sum_{\ell \in \mathcal{L}(r)} \Pr\{1 \in Z_\ell(t)\} = O(\delta^{3/2}).$$

We turn to bounding the probability that $[m] \subseteq Z(t)$. We fix $\ell^{(1)},\ldots,\ell^{(m)} \in \mathcal{L}$ and bound the probability that $i \in Z_{\ell^{(i)}}(t)$ for $i = 1,\ldots,m$. Let $k(i) = \dim \ell^{(i)}$ denote the number of components of $\ell^{(i)}$. Then, summing over all choices $s_j^{(i)}$ for $1 \leq i \leq m$ and $1 \leq j \leq k(i)$,

$$\begin{aligned}
&\Pr\{i \in Z_{\ell^{(i)}}(t), i = 1,\ldots,m\} \\
&\leq \sum_{s_j^{(i)}} \prod_{i=1}^{m} B^{k(i)} \frac{(rn-2t)^{(r-\sum_j \ell_j^{(i)})/2}}{(rn)^{r/2}} \prod_{j=1}^{k(i)} (rn-2s_j^{(i)})^{\ell_j^{(i)}/2-1} \\
&\leq \prod_{i=1}^{m} \left[B^{k(i)} \frac{(rn-2t)^{(r-\sum_j \ell_j^{(i)})/2}}{(rn)^{r/2}} \prod_{j=1}^{k(i)} \left(\sum_{s=0}^{t} (rn-2s)^{\ell_j^{(i)}/2-1} \right) \right] \\
&\leq B^{\sum k(i)} \delta^{3m/2} = O((B^r\delta)^{3m/2}).
\end{aligned}$$

Summing over all $O(m)$ choices of $\ell^{(i)}, i = 1,\ldots,m$, we have

$$\Pr\{[m] \subseteq Z(t)\} = O(m(B^r\delta)^{3m/2}) \leq (C\delta)^{3m/2}$$

for some constant $C > 0$. By symmetry the same bound holds for any vertex set of size m. It follows that for any m, writing $(n)_m = n(n-1)\ldots(n-m+1)$,

$$\mathbb{E}\left((Z(t))_m\right) \leq (n)_m \times (C\delta)^{3m/2} \leq (Cn\delta^{3/2})^m.$$

For $s > 1$ we apply the binomial theorem to obtain

$$\mathbb{E}\left(s^{Z(t)}\right) = \mathbb{E}\left((1+(s-1))^{Z(t)}\right) = \sum_{m \geq 0} \frac{\mathbb{E}\left((Z(t))_m\right)(s-1)^m}{m!}.$$

We set $s = e^\theta \leq 1+2\theta$ (as $\theta = o(1)$) to obtain

$$\mathbb{E}\left(e^{\theta Z(t)}\right) \leq \sum_{m \geq 0} \frac{(Cn\delta^{3/2})^m (2\theta)^m}{m!} \leq \exp\left\{\theta D n \delta^{3/2}\right\},$$

for some $D > 0$. ◀

▶ **Corollary 13.** *For $t = (1-\delta)\frac{rn}{2}$ with $\delta = o(1)$, and $0 < \theta = o(1)$,*

$$\mathbb{E}\left(e^{-\theta X_1^g(t)}\right) = \exp\{-\theta rn\delta(1-o(1))\}$$

The technique used to prove Lemma 12 can be strengthened to obtain concentration for the number of unvisited vertices $X_r(t)$.

▶ **Lemma 14.** *For $\theta > 0$,*

$$\mathbb{E}\left(e^{\theta X_r(t)}\right) \leq \exp\left\{2\theta n \delta^{r/2}\right\}. \tag{11}$$

Furthermore, if $t = (1-\delta)\frac{rn}{2}$ *with* $\delta = o(1)$ *and* $n\delta^{r/2} \to \infty$, *then for any* ω *tending to infinity arbitrarily slowly,*

$$Pr\left\{|X_r(t) - n\delta^{r/2}| > \frac{n\delta^{r/2}}{\omega^{1/2}}\right\} \leq \frac{1}{\omega}.$$

Finally, if $n\delta^{r/2} = o(1)$ *then* $X_r(t) = 0$ *w.h.p.*

Lemma 14, the proof of which is omitted here, relates the number of unvisited edges to the number of unvisited vertices: we expect $|X_r(t)| = n - s$ to occur when $t \approx \left(1 - \frac{s}{n}\right)^{2/r}$. This heuristically explains why $C_E^b(G_r) \sim \frac{r}{2} C_V^b(G_r)$. Detailed calculations for the vertex cover time are carried out for $r = 3$ in [7], and the calculations for larger r are identical.

B The green edges

Let $\Phi(t)$ denote the number of green edges in $W(t)$.

▶ **Lemma 15.** *Let* $0 < \varepsilon < r - 2$ *and define*

$$\delta_\varepsilon = \left(\frac{\log^4 n}{n}\right)^{\frac{r-1}{r+\varepsilon}}, \quad t_\varepsilon = (1-\delta_\varepsilon)\frac{rn}{2}.$$

Then with high probability, $\Phi(t) \geq n\delta^{\frac{1+\varepsilon}{r-1}}$ *for all* $t_1 \leq t \leq t_\varepsilon$.

Proof. Firstly, let us see how $\Phi(t)$ changes with time. Fix $\varepsilon_1 > 0$ such that

$$\frac{1}{(1-\varepsilon_1)(r-1)} < \frac{1+\varepsilon}{r-1}, \tag{12}$$

and let

$$\mathcal{X}(t) = \{X_1^g(t) \geq (1-\varepsilon_1)(rn - 2t)\}$$

and let $\mathbf{1}_t$ denote the indicator variable for $\mathcal{X}(t)$. We note that with $\lambda = 1/\log n$, by Corollary 13

$$\Pr\left\{\overline{\mathcal{X}(t)}\right\} \leq \frac{\mathbb{E}\left(e^{-\lambda X_1^g(t)}\right)}{e^{-\lambda(1-\varepsilon_1)(rn-2t)}} \leq \exp\left\{-\frac{\varepsilon_1 n\delta_\varepsilon}{\log n}\right\} =: \eta, \tag{13}$$

for any $t \leq t_\varepsilon$.

▶ Claim 1. For $0 < \theta \leq \delta_\varepsilon \log^{-2} n$, $\varepsilon_1 > 0$ and $t_0 \leq t \leq t_\varepsilon$,

$$\mathbb{E}\left(e^{-\theta(\Phi(t+1)-\Phi(t))}\mathbf{1}_t \mid [W(t)]\right) \leq \exp\left\{\frac{2\theta\Phi(t)}{(1-\varepsilon_1)(r-1)(rn-2t)}(1+O(\gamma))\right\}\mathbf{1}_t,$$

with $\gamma = o(\log^{-1} n)$.

Proof of Claim 1. Condition on a $[W(t)]$ such that $X_1^g(t) \geq (1-\varepsilon_1)(rn-2t)$. If the next edge is added without entering a blue walk, then $\Phi(t+1) = \Phi(t) + 1$. So,

$$\Pr\{\Phi(t+1) - \Phi(t) = 1 \mid [W(t)]\} = 1 - \frac{X_1(t)}{rn-2t}.$$

Suppose the new edge chooses a vertex of $X_1(t)$, thus entering a blue walk. We may view this as a walk on $[W(t)]$, and any time a green edge is traversed, we ask if the green edge

in $[W(t)]$ contains a green link in $W(t)$, in which case the blue walk ends. If not, the green edge turns blue and Φ decreases by one.

There are $L(t) = \frac{r-1}{2}X_1^g(t)$ green links, distributed into the $\Phi(t)$ green edges by a Pólya urn process as discussed in Section 5. Suppose $e_1, e_2, \dots, e_\ell$ are green edges in $[W(t)]$, and let $K_1, K_2, \dots, K_\ell$ be the lengths of the corresponding paths in $W(t)$, corresponding to the first ℓ entries of a vector $(k_1, \dots, k_\phi)$ drawn uniformly at random from all vectors with $k_i \geq 1$ and $\sum_{i=1}^{\phi} k_i = \Phi(t)$. The probability that none of the ℓ edges contains a green link is exactly

$$\Pr\{K_i = 1 \text{ for } i = 1, 2, \dots, \ell\} = \prod_{i=1}^{\ell} \frac{\binom{\Phi-i-1}{\phi-i-1}}{\binom{\Phi-i}{\phi-i}} = \prod_{i=1}^{\ell}\left(1 - \frac{L(t)}{\Phi(t)-i}\right) \leq \left(1 - \frac{L(t)}{\Phi(t)}\right)^{\ell}.$$

This shows that the number of green edges visited before discovering a green link can be bounded by a geometric random variable. If a green edge is visited without a discovery, that edge turns blue. Note that the blue walk may also end when a vertex of X_i^b is found for some $i \geq 1$; we are upper bounding the number of green edges visited.

So in distribution,

$$\Phi(t+1) - \Phi(t) \stackrel{d}{=} 1 - B\left(\frac{X_1(t)}{rn-2t}\right) R_t$$

where $B(p)$ denotes a Bernoulli random variable taking value 1 with probability p, and R_t is stochastically dominated above by a geometric random variable with success probability $L(t)/\Phi(t)$. The two random variables on the right-hand side are independent. So

$$\mathbb{E}\left(e^{-\theta(\Phi(t+1)-\Phi(t))} \mid [W(t)]\right) = e^{-\theta}\left(1 - \frac{X_1(t)}{rn-2t} + \frac{X_1(t)}{rn-2t}\mathbb{E}\left(e^{\theta R_t} \mid [W(t)]\right)\right)$$

The map $x \mapsto e^{\theta x}$ is increasing for $\theta > 0$, so we can couple R_t to a geometric random variable S_t with success probability $L(t)/\Phi(t)$ in such a way that

$$\mathbb{E}\left(e^{\theta R_t} \mid [W(t)]\right) \leq \mathbb{E}\left(e^{\theta S_t} \mid [W(t)]\right).$$

As S_t is geometrically distributed and $X_1^g(t) \geq (rn-2t)/2$ by conditioning on $\mathcal{X}(t)$,

$$\mathbb{E}\left(e^{\theta S_t} \;\middle|\; [W(t)]\right) = 1 + \theta\frac{\Phi(t)}{L(t)} - O\left(\frac{\theta^2\Phi(t)^2}{L(t)^2}\right) = 1 + \theta\frac{\Phi(t)}{L(t)}(1 + O(\gamma)).$$

Conditioning on $X_1^g(t) \geq (1-\varepsilon_1)(rn-2t)$ implies that $L(t) = \frac{r-1}{2}X_1^g(t) = \Omega(n\delta)$, so

$$\gamma := \theta\frac{\Phi(t)}{L(t)} \leq \delta_\varepsilon \log^{-2} n \frac{n}{\Omega(n\delta_\varepsilon)} = o(\log^{-1} n).$$

We also have $X_1^b(t) \leq rn - 2t - X_1^g(t)$, so

$$\frac{X_1(t)}{L(t)} = \frac{X_1^g(t)}{L(t)} + \frac{X_1^b(t)}{L(t)} \leq \frac{2}{r-1} + \frac{\varepsilon_1(rn-2t)}{(1-\varepsilon_1)\frac{r-1}{2}(rn-2t)} = \frac{2}{(1-\varepsilon_1)(r-1)}.$$

So for $[W(t)] \in \mathcal{X}(t)$,

$$\begin{aligned}
&\mathbb{E}\left(e^{-\theta(\Phi(t+1)-\Phi(t))}\mathbf{1}_t \mid [W(t)]\right) \\
&\leq e^{-\theta}\left(1 - \frac{X_1(t)}{rn-2t} + \frac{X_1(t)}{rn-2t}\left(1 + \theta\frac{\Phi(t)}{L(t)}(1+O(\gamma))\right)\right) \\
&\leq \exp\left\{\frac{2\theta\Phi(t)}{(1-\varepsilon_1)(r-1)(rn-2t)}(1+O(\gamma))\right\}.
\end{aligned}$$

◀

Define for $0 < \theta = o(1)$,

$$f_t(\theta) = \mathbb{E}\left(e^{-\theta\Phi(t)}\mathbf{1}_t\right).$$

As $\Phi(t) \geq L(t) = \frac{r-1}{2}X_1^g(t)$ we have for $0 < \theta = o(1)$, by Corollary 13,

$$f_{t_0}(\theta) \leq \mathbb{E}\left(e^{-\theta\Phi(t_0)}\right) \leq \mathbb{E}\left(e^{-\theta\frac{r-1}{2}X_1^g(t)}\right) = \exp\left\{-\theta\frac{r-1}{2}rn\delta_0(1+o(1))\right\}. \tag{14}$$

Claim 1 shows that for $t_0 \leq t < t_\varepsilon$,

$$f_{t+1}(\theta) \leq f_t\left(\theta\left(1 - \frac{2(1+O(\gamma))}{(1-\varepsilon_1)(r-1)(rn-2t)}\right)\right) + \eta$$

where $\eta = \exp\{-\varepsilon_1 n\delta_\varepsilon/\log n\}$ is an upper bound for $\Pr\left\{\overline{\mathcal{X}(t+1)}\right\}$, as defined in (13). As $\gamma = o(\log^{-1} n)$, we have

$$\prod_{s=t_0}^{t-1}\left(1 - \frac{2(1+O(\gamma))}{(1-\varepsilon_1)(r-1)(rn-2s)}\right) \sim \left(\frac{rn-2t}{rn-2t_0}\right)^{\frac{1}{(1-\varepsilon_1)(r-1)}}.$$

It follows by induction and from (14) that if $F(t) = n\delta^{\frac{1+\varepsilon}{r-1}}$,

$$\begin{aligned}
f_t(\theta) &\leq f_{t_0}\left(\theta\prod_{s=t_0}^{t-1}\left(1 - \frac{2(1+O(\gamma))}{(1-\varepsilon_1)(r-1)(rn-2s)}\right)\right) + (t-t_0)\eta \\
&\leq \exp\left\{-\theta rn\delta_0\left(\frac{\delta}{\delta_0}\right)^{\frac{1}{(1-\varepsilon_1)(r-1)}}\right\} + (t-t_0)\eta \\
&\leq \exp\left\{-r\theta F(t)\right\} + n\eta.
\end{aligned}$$

Here we used the fact that ε_1 was chosen in (12) to satisfy $1/(1-\varepsilon_1)(r-1) < (1+\varepsilon)/(r-1)$.

Now, setting $\theta = \delta_\varepsilon \log^{-2} n$, using the bound $\mathbf{1}_{\{X>a\}} \leq X/a$,

$$\begin{aligned}
\Pr\left\{\Phi(t) < F(t)\right\} &\leq \Pr\left\{\overline{\mathcal{X}(t)}\right\} + \Pr\left\{\Phi(t) < F(t),\ \mathcal{X}(t)\right\} \\
&\leq \eta + \mathbb{E}\left(\mathbf{1}_{\left\{e^{-\theta\Phi(t)} > e^{-\theta F(t)}\right\}}\mathbf{1}_t\right) \\
&\leq \eta + e^{\theta F(t)} f_t(\theta) \\
&= O(ne^{\theta F(t)}\eta) + e^{-\theta(r-1)F(t)} \\
&= o(n^{-1}).
\end{aligned}$$

It follows that $\Phi(t) \geq F(t)$ for all t in the given range with high probability. ◀

Satisfiability and Derandomization for Small Polynomial Threshold Circuits

Valentine Kabanets
School of Computing Science, Simon Fraser University, Burnaby, BC, Canada
kabanets@sfu.ca

Zhenjian Lu
School of Computing Science, Simon Fraser University, Burnaby, BC, Canada
zla54@sfu.ca

Abstract

A polynomial threshold function (PTF) is defined as the sign of a polynomial $p\colon \{0,1\}^n \to \mathbb{R}$. A PTF circuit is a Boolean circuit whose gates are PTFs. We study the problems of exact and (promise) approximate counting for PTF circuits of constant depth.

- **Satisfiability (#SAT).** We give the first zero-error randomized algorithm faster than exhaustive search that counts the number of satisfying assignments of a given constant-depth circuit with a super-linear number of wires whose gates are s-sparse PTFs, for s almost quadratic in the input size of the circuit; here a PTF is called s-sparse if its underlying polynomial has at most s monomials. More specifically, we show that, for any large enough constant c, given a depth-d circuit with $(n^{2-1/c})$-sparse PTF gates that has at most $n^{1+\varepsilon_d}$ wires, where ε_d depends only on c and d, the number of satisfying assignments of the circuit can be computed in randomized time $2^{n-n^{\varepsilon_d}}$ with zero error. This generalizes the result by Chen, Santhanam and Srinivasan (CCC, 2016) who gave a SAT algorithm for constant-depth circuits of super-linear wire complexity with linear threshold function (LTF) gates only.
- **Quantified derandomization.** The quantified derandomization problem, introduced by Goldreich and Wigderson (STOC, 2014), asks to compute the majority value of a given Boolean circuit, under the promise that the minority-value inputs to the circuit are very few. We give a quantified derandomization algorithm for constant-depth PTF circuits with a super-linear number of wires that runs in quasi-polynomial time. More specifically, we show that for any sufficiently large constant c, there is an algorithm that, given a degree-Δ PTF circuit C of depth d with n^{1+1/c^d} wires such that C has at most $2^{n^{1-1/c}}$ minority-value inputs, runs in quasi-polynomial time $\exp\left((\log n)^{O(\Delta^2)}\right)$ and determines the majority value of C. (We obtain a similar quantified derandomization result for PTF circuits with n^{Δ}-sparse PTF gates.) This extends the recent result of Tell (STOC, 2018) for constant-depth LTF circuits of super-linear wire complexity.
- **Pseudorandom generators.** We show how the classical Nisan-Wigderson (NW) generator (JCSS, 1994) yields a nontrivial pseudorandom generator for PTF circuits (of unrestricted depth) with sub-linearly many gates. As a corollary, we get a PRG for degree-Δ PTFs with the seed length $\exp\left(\sqrt{\Delta \cdot \log n}\right) \cdot \log^2(1/\epsilon)$.

2012 ACM Subject Classification Theory of computation → Circuit complexity

Keywords and phrases constant-depth circuits, polynomial threshold functions, circuit analysis algorithms, SAT, derandomization, quantified derandomization, pseudorandom generators.

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.46

Related Version A full version of this paper is available at `https://eccc.weizmann.ac.il/report/2018/115` as ECCC TR18-115.

Acknowledgements We thank Suguru Tamaki for suggesting to us to consider sparse PTFs in the case of satisfiability, and for clarifying the MAX-k-SAT algorithm in [20] for us.

1 Introduction

Satisfiability and derandomization are famous examples of "circuit analysis" problems that, apart from being important algorithmic problems in their own right, are also intimately related to the notoriously difficult problem of proving circuit lower bounds. In this paper, we give several algorithmic results for these problems for the class of Boolean circuits with polynomial-threshold functions (PTFs) as gates.

Circuit-SAT. Circuit-SAT asks to determine whether a given Boolean circuit has a satisfying assignment. As a canonical NP-complete problem, it is not believed to have a polynomial-time (or subexponential-time) algorithm. However, it is still very interesting to look for nontrivial algorithms for Circuit-SAT running faster than naive exhaustive search. More specifically, given a circuit of polynomial size on n variables, is there a satisfiability algorithm that runs in time at most $2^n/n^{\omega(1)}$?

It turns out that this task is challenging even for very restricted classes of circuits. The difficulty of obtaining such a SAT algorithm can be partially explained by the work of Williams [26, 27] showing that a Circuit-SAT algorithm faster than exhaustive search for a given class of circuits can often be used to prove nontrivial circuit lower bounds against that same class of circuits (given that the class of circuits satisfies some mild conditions). In fact, Williams designed such a Circuit-SAT algorithm for ACC circuits (constant-depth circuits with AND, OR, NOT, and modular counting gates) that runs in time $2^{n-n^{1/\exp(d)}}$ (recently improved to $2^{n-n^{1/\mathrm{poly}(d)}}$ by [6]), where d is the depth of the circuit, and then used this algorithm to show that NEXP contains a language that is not computable by any family of polynomial-size constant-depth ACC circuits, a breakthrough result in circuit complexity.

Given the connections between nontrivial Circuit-SAT algorithms and circuit lower bounds, one of the next big goals in circuit complexity is to design such an algorithm for the class of TC^0 circuits, constant-depth circuits with majority gates. Lower bounds against the class of polynomial-size TC^0 circuits is currently one of the most important open problems in complexity.

Derandomization. A central problem in derandomization is to give an efficient deterministic algorithm for computing the majority value of a given Boolean circuit, under the promise that the fraction of minority-value inputs to the circuit is at most $1/3$. That is, given a circuit that outputs some unknown value $b \in \{0,1\}$ on all but at most $1/3$ fraction of inputs, we need to determine this majority value b, efficiently deterministically.

As for Circuit-SAT, it is also known that a "faster-than-brute-force" algorithm solving the aforementioned derandomization problem for a circuit class $\mathcal{C}$ (satisfying some mild conditions) implies lower bounds against that class $\mathcal{C}$ [26].

Black-Box Derandomization: Pseudorandom generators. One way to solve the derandomization problem for a class $\mathcal{C}$ of circuits is to construct a pseudorandom generator (PRG) for $\mathcal{C}$. A PRG for a class $\mathcal{C}$ of n-input Boolean circuits is an efficiently deterministically computable function G mapping short binary strings (seeds) to longer binary strings so that every $C \in \mathcal{C}$ accepts G's output on a uniformly random seed with about the same probability as that for an actual uniformly random string. More precisely, we say that a generator $G\colon \{0,1\}^r \to \{0,1\}^n$ is *ϵ-fooling* for a class $\mathcal{C}$ of Boolean circuits if for every $C\colon \{0,1\}^n \to \{0,1\}$ from $\mathcal{C}$, $|\mathbf{Pr}[C(G(x)) = 1] - \mathbf{Pr}[C(y) = 1]| \leq \epsilon$, for uniformly random $x \in \{0,1\}^r$ and $y \in \{0,1\}^n$. The parameter r is called the seed length of the PRG. Then

given a PRG that fools $\mathcal{C}$, for every $C \in \mathcal{C}$, we can estimate the fraction of accepted inputs to within an additive error ϵ, by trying all possibles seeds. This gives a deterministic algorithm solving the derandomization problem in time approximately 2^r.

Note that a PRG yields *black-box* derandomization in the sense that we do not need to be given as input a circuit $C \in \mathcal{C}$ in order to decide the set of 2^r query points for C; the set of 2^r query points is the same for all circuits in class $\mathcal{C}$.

Quantified derandomization. As standard derandomization appears difficult even for weak circuit classes, one considers relaxations. One relaxation is to assume that a given n-input circuit C outputs an unknown value $b \in \{0,1\}$ on all but "very few" inputs, e.g., $2^n/n^{\omega(1)}$ inputs rather than $2^n/3$ in the case of standard derandomization. Goldreich and Wigderson [8] named this a *quantified derandomization problem*. More formally, for a class $\mathcal{C}$ of circuits, and a function $B\colon \mathbb{N} \to \mathbb{N}$, the $(\mathcal{C}, B)$-quantified derandomization problem is the following: given a circuit $C \in \mathcal{C}$ such that C has at most $B(n)$ minority-value inputs in $\{0,1\}^n$, determine the majority value $b \in \{0,1\}$ for C.

It was immediately observed by [8] that for "sufficiently powerful" circuit classes (e.g., $\mathsf{AC}^0[\oplus]$, polynomial-size constant-depth circuits with unbounded fan-in AND, OR, parity gates, and negation gates), quantified derandomization is *equivalent* to standard derandomization, as one can perform efficient pseudo-random sampling (via randomness extractors) within the same circuit class. Thus, quantified derandomization may be possible to achieve (given our current knowledge) only for "very weak" circuit classes. [8] gave quantified derandomization algorithms for AC^0 (later strengthened by [22]) and some other classes. Recently, Tell [24] showed that quantified derandomization is also possible for constant-depth LTF circuits of small super-linear wire complexity (and that improving this to slightly higher super-linear wire complexity is as hard as getting nontrivial standard derandomization for the circuit class TC^0, which in turn would imply TC^0 circuit lower bounds).

PTF circuits. The focus of the present paper is on circuits whose gates are polynomial threshold functions. An n-variate polynomial threshold function (PTF) is defined as the sign $\mathrm{sgn}(p)$ of a multi-linear polynomial $p\colon \{0,1\}^n \to \mathbb{R}$. Here, for $v \in \mathbb{R}$, we define the sign function $\mathrm{sgn}(v)$ to be 1 on $v > 0$, and 0 on $v < 0$. There are two common complexity measures for PTFs: *degree*, which is the degree of p, and *sparsity*, which is the number of monomials in p, where a monomial is of the form $\prod_{i \in S}(x_i \oplus b_i)$ where $S \subseteq [n]$ and $b_i \in \{0,1\}$ for each $i \in S$. We call the PTF s-sparse if $p(x_1, \ldots, x_n)$ is the sum of at most s monomials. PTFs of degree 1 are called linear threshold functions (LTFs). Thus an s-sparse PTF can be equivalently defined as an LTF of at most s terms, where each term is an AND of literals (variables and their negations).

Polynomial threshold circuits are circuits whose gates are PTFs. We will study both circuits with low-degree PTF gates and circuits with sparse PTF gates. We call a circuit degree-Δ PTF circuit if its gates are degree-Δ PTFs. Similarly, a circuit is called s-sparse PTF circuit if its gates are s-sparse PTFs. We note that when discussing circuits, the word "sparse" is often used to describe circuits with a small number of wires (recall that the number of wires is the sum of fan-ins over all gates of the circuit). To avoid ambiguity, we clarify that in this paper the word "sparse" always refers to PTFs. For example, a sub-quadratically sparse PTF circuit means a circuit with gates that are sub-quadratically sparse PTFs (i.e., PTFs that have a sub-quadratic number of monomials).

1.1 Our results

Circuit-SAT for sub-quadratically sparse PTF circuits with $n^{1+\varepsilon}$ wires. PTFs are very powerful even for small sparsity. For example, s-sparse PTFs can encode MAX-SAT with s clauses and exponential weights, a problem known how to solve nontrivially only for a sub-quadratic number of clauses. Therefore, a nontrivial SAT algorithm for PTFs of quadratic sparsity would break the current barrier of solving MAX-SAT with exponential weights. In fact, since a polynomial of degree-2 has at most a quadratic number of monomials, such an algorithm would also give a nontrivial SAT algorithm for degree-2 PTFs, which is currently unknown[1].

We give the first nontrivial #SAT algorithms (counting the number of satisfying assignments of a given circuit) for the class of constant-depth circuits with PTF gates, where the PTF circuit has small super-linear wire complexity (defined as the sum of fan-ins over all gates of the circuit) and each PTF gate has sub-quadratic sparsity. Our main result is the following.

▶ **Theorem 1** (#SAT algorithm for sub-quadratically sparse PTF circuits)**.** *There is a constant $b_1 > 1$ such that, for every $c \geq b_1$ and $d > 0$, there is a zero-error randomized algorithm that counts the number of satisfying assignments of any given depth-d, n-variate circuit with*

- *$(n^{2-1/c})$-sparse PTF gates, and*
- *at most $n^{1+\varepsilon_d}$ wires.*

The running time of this #SAT algorithm is at most $2^{n-n^{\varepsilon_d}}$, where $\varepsilon_d = c^{-3^d}$.

We also get an algorithm with better parameters if we further assume that the sparse PTF gates in the circuit have low degree. Let $\mathcal{G}_{\Delta,c}$ denote the class of Boolean functions where each function can be computed as an LTF of at most $n^{2-1/(c\cdot\Delta^2)}$ arbitrary Δ-variate Boolean functions.

▶ **Theorem 2** (#SAT algorithm for sub-quadratically sparse PTF circuits with low-degree)**.** *There exists a constant $b_2 > 1$ such that, for every $d, \Delta > 0$ and $c \geq b_2$, there is a zero-error randomized algorithm that counts the number of satisfying assignments of any given depth-d, n-variate circuit with*

- *gates from $\mathcal{G}_{\Delta,c}$, and*
- *at most $n^{1+\varepsilon_{d,\Delta}}$ wires.*

The running time of this #SAT algorithm is at most $2^{n-n^{\varepsilon_{d,\Delta}}}$, where $\varepsilon_{d,\Delta} = \left(c\cdot\Delta^2\right)^{-d}$.

Quantified derandomization for PTF circuits with $n^{1+\varepsilon}$ wires in quasi-polynomial time.

▶ **Theorem 3** (Quantified derandomization for low-degree (or sparse) PTF circuits)**.** *For any constant $c \geq 122$ and any $\Delta, d > 0$ such that $\Delta \ll \sqrt{\log n/(c^d \cdot \log\log n)}$, let $\mathcal{C} = \mathcal{C}(n, d, \Delta, c)$ be the class of n-variate, depth d PTF circuits with*

- *degree-Δ PTF gates (or n^{Δ/c^d}-sparse PTF gates), and*
- *at most n^{1+1/c^d} wires.*

The $\left(\mathcal{C}, 2^{n^{1-7/\sqrt{c}}}\right)$-quantified derandomization problem is solvable in time $2^{(\log n)^{O\left(\Delta^2\right)}}$.

---

[1] Sakai, Seto, Tamaki and Teruyama [20] recently reported a faster-than-brute-force algorithm for MAX-k-SAT for any constant k with arbitrary weights (which implies a satisfiability algorithm for degree-k PTFs). However, their algorithm is conditional in that it relies on an assumption that one can *efficiently* reduce the weights of a given n-variate LTF to integral weights of magnitude at most $2^{O(n\log n)}$. While it is known that such small weights exist for every LTF [17], it is currently not known how to find them efficiently.

PRG for PTF circuits with few gates.

▶ **Theorem 4** (PRG for PTF Circuits). *There exists a constant $E > 0$ such that the following holds. For any positive integers α and Δ, let $\mathcal{C} = \mathcal{C}(n, \alpha, \Delta)$ be the class of degree-Δ PTF circuits on n inputs with at most $s = n^{\frac{1}{\alpha+1}} / \left(E \cdot 5^{\alpha \cdot \Delta} \cdot \log^2(n) \cdot \log(n/\epsilon)\right)$ gates. There exists a $\mathsf{poly}(n)$-time computable PRG $G\colon \{0,1\}^r \to \{0,1\}^n$ ϵ-fooling $\mathcal{C}$, where the seed length is $r = n^{2/(\alpha+1)}$.*

We get the following PRG for a single PTF (by setting α appropriately).

▶ **Corollary 5** (PRG for PTFs). *There exists a PRG $G\colon \{0,1\}^r \to \{0,1\}^n$, computable in deterministic time $\mathsf{poly}(n)$, that ϵ-fools degree-Δ PTFs on n variables with the seed length $r = \exp\left(O\left(\sqrt{\Delta \cdot \log n}\right)\right) \cdot \log^2(1/\epsilon)$.*

1.2 Our techniques

A common way to analyze constant-depth circuits is to apply (random) restrictions, getting some depth reduction, and iterate, until the resulting circuit becomes very simple. Our #SAT algorithms and quantified derandomization algorithms for constant-depth PTF circuits also follow this approach, mainly relying on the ideas of [5] for depth reduction, and [11] for (pseudo-) random restrictions for PTFs. Our PRG is based on the celebrated Nisan-Wigderson "hardness-based" generator (NW PRG) [19]. We give more details in the following sections as we discuss each of the results.

1.3 Related work and comparison

Circuit Satisfiability. Impagliazzo, Paturi and Schneider [10] gave a Circuit-SAT algorithm for depth-2 LTF circuits with few wires; this result was improved by Chen and Santhanam [4]. Recently, Alman, Chan and Williams [1] and Tamaki [21] both gave Circuit-SAT algorithms for depth-2 LTF circuits with an almost quadratic number of gates.

The most closely related previous work is by Chen, Santhanam and Srinivasan [5] who gave a Circuit-SAT algorithm for circuits with a super-linear number of wires whose gates are LTFs. In particular, they show that the satisfiability of a depth-d, n-variate circuit with LTF gates and at most $n^{1+\varepsilon_d}$ wires can be solved by a zero-error randomized algorithm in time $2^{n-n^{\varepsilon_d}}$, where $\varepsilon_d = c^{-d}$ for some constant c. Our results extend their algorithm to the more general case of circuits with *sparse PTF* gates. In particular, our algorithm in Theorem 2 for $\Delta = 1$ subsumes the Circuit-SAT algorithm for LTFs in [5]. Also note that the sparsity of the PTF gates in our model is almost quadratic in n, which is the input size of the circuit. "Opening up" the PTF gates in the circuit and expressing them as LTFs of terms will result in a (constant-depth) LTF circuit that can have an almost quadratic number of wires, and such a circuit cannot be analyzed by the result in [5].

Quantified derandomization. The quantified derandomization problem was first introduced by Goldreich and Wigderson in [8], where they obtained a polynomial time algorithm that finds the majority output of a given AC^0 circuit that has at most $2^{n^{0.999}}$ minority-value inputs. The key tool in their algorithm is a derandomized version of Håstad's switching lemma [9] with logarithmic seed length. In addition, they obtain quantified derandomization results for log-space algorithms and arithmetic circuits. The quantified derandomization algorithm for AC^0 was generalized by Tell [22] to handle AC^0 circuits with at most $2^{\Omega(n/\log^{d-2} n)}$ minority-value inputs, where d is the depth, with an increase of the running time to $2^{\tilde{O}(\log^3 n)}$. As

mentioned above, Tell [24] has recently obtained a quantified derandomization algorithm for depth-d LTF circuits with $n^{1+1/\exp(d)}$ wires with at most $2^{n^{1-1/5d}}$ minority-value inputs, running in time $n^{(\log\log n)^2}$. Our result extends this to low-degree PTF circuits and sparse PTF circuits, at the expense of increasing the running time to quasi-polynomial (for constant degree and polynomial sparsity). For the results on reducing standard derandomization to quantified derandomization, see [8, 22, 23, 24].

PRGs. There has been a long sequence of works on constructing PRGs (of varying strength) for various sub-classes of $\mathsf{P/poly}$. Among these known PRG constructions, some are NW-style "hardness-based" generators, while others are *ad hoc* constructions (often using such standard pseudorandomness tools as hashing, limited-wise independence, expander graphs, etc.) The previous PRGs for PTFs due to [16, 13] are of the latter kind. The construction uses hashing and limited-wise independence. The analysis is quite involved, and depends on a number of analytic tools for polynomials (concentration and anti-concentration results, the invariance principle, hypercontractivity, regularization, etc.). In contrast, our PRG for PTFs (of Corollary 5) is the NW-style construction, whose analysis is simple, assuming an average-case lower bound for an appropriate class of functions.

For constant degree PTFs and constant error ϵ, the PRG of [16, 13] has exponential stretch (mapping a seed of length $O(\log n)$ to an n-bit string fooling n-input PTFs). However, these PRGs has polynomial dependence in the error $1/\epsilon$ and cannot handle small error. Our PRG cannot achieve such exponentially long stretch for constant error, but it can achieve even exponentially small error ϵ with a nontrivial (sub-linear) seed size, which is impossible for the PRGs of [16, 13].

In their work studying correlation bounds for AC^0 circuits with few symmetric gates [14], Lovett and Srinivasan obtained an average-case hard function for constant depth poly-size AC^0 circuits with few LTF gates and used it to construct a PRG fooling such circuits with polynomial stretch and exponentially small error, also based on the generic construction of Nisan and Wigderson. Since a PTF can be viewed as a depth-2 circuit computing an LTF of ANDs, such a PRG also fools small PTF circuits. While the PRG in [14] can fool a more general model, which is constant-depth AC^0 circuits augmented with LTF gates, it can have only polynomial seed stretch and the circuit can have only constant depth. Our work here focuses on circuits with only PTF gates. Our PRG can have sub-polynomial seed length and it can fool PTF circuits regardless of the depth as long as the number of gates is small. In particular, our PRG for a single PTF with sub-polynomial seed length (Corollary 5) can be used to construct a PRG for degree-2 PTFs with a seed length that is logarithmic in the input size and *sub-polynomial* in the error (see [12]).

Threshold circuits. It is well known that the class of constant-depth polynomial-size TC^0 circuits is equivalent to the class of constant-depth polynomial-size circuits with LTF gates [7]. LTF circuits have been intensively studied in complexity theory. PTF circuits have been previously studied for lower bounds [18, 11]. Threshold circuits are also studied as a model of artificial neural networks [15] (see also [2]), where a threshold gate is also called a neuron.

Remainder of the paper. We give the necessary background in Section 2. We prove our satisfiability algorithms (Theorem 1). We describe our quantified derandomization result for low-degree PTF Circuits in Section 4, and our PRG result in Section 5. We conclude with some open problems in Section 6. For lack of space, we omit many proofs in this extended abstract and refer the reader to the full version for those proofs.

2 Preliminaries

2.1 Notation

For a positive integer n, let $[n]$ denote the set $\{1, 2, \ldots, n\}$. For a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$, we define the *majority value* of f to be the bit value $b \in \{0,1\}$ that maximizes the quantity $\mathbf{Pr}_{x\sim\{0,1\}^n}[f(x) = b]$, and we call $1-b$ the *minority value*. We say that two Boolean functions f and g are δ-*close* if $\mathbf{Pr}_x[f(x) \neq g(x)] \leq \delta$. We say that a function f is δ-close to an *explicit* constant if f is δ-close to some constant function and such a constant function can be efficiently determined from f.

We will often view an s-sparse PTF as an LTF of at most s AND gates. It is well known that every LTF on m variables has a canonical representation, where the coefficients are integers of magnitude at most $2^{O(m \log m)}$ [17]. Therefore, every s-sparse PTF is equivalent to some s-sparse PTF whose coefficients are integers of magnitude at most $2^{O(s \log s)}$. Without loss of generality, for a circuit with s-sparse PTF gates, we assume the coefficients of all gates have bit complexity $\mathsf{poly}(s)$.

2.2 Random restrictions

A random restriction is a process that randomly fixes the values of a subset of variables. We will often view a random restriction as a two-step process: the first step is selecting (in some random manner) a subset of unrestricted variables and the second step is fixing (in some random manner) the values of all the other variables. Depending on different contexts, we will consider different types of random restrictions based on how the unrestricted variables are picked and how the restricted variables are fixed.

Truly random restriction. The first type is the *(truly) r-random restriction*, for a parameter $0 < r < 1$. It is the process that leaves each variable, independently, free with probability r, and otherwise assigns it 0 or 1 uniformly at random.

Pseudorandom restriction using limited-wise independence. We can also pick and fix variables in a pseudorandom manner. One way to do this is to use a limited-wise independent distribution. For integers $n, m > 0$, a distribution X on $[m]^n$ is called k-wise independent if any k coordinates of X are uniformly distributed. That is, for any $1 \leq i_1, \ldots, i_k \leq n$ and every $b_1, \ldots, b_k \in [m]$, we have $\mathbf{Pr}[X_{i_1} = b_1, \ldots, X_{i_k} = b_k] = m^{-k}$. A k-wise independent distribution over $[m]^n$ can be constructed using $k \cdot \log n$ random bits for $m \leq n$ (see, e.g., [25]).

For a random restriction ρ, we say that ρ picks the unrestricted variables k-wise independently, each with probability r, if each variable is set to be unrestricted by ρ with probability r, and any k of the variables are independent. Note that this process can be done using a k-wise independent distribution over $[1/r]^n$, where n is the number of variables. Also, we say that ρ fixes the variables k-wise independently if each variable is assigned 0 or 1 uniformly at random by ρ and any k of the variables are independent.

Random block restriction. A random block restriction picks the set of unrestricted variables by picking a block from some arbitrary predetermined partition of variables. More formally, an *m-block random restriction* for a function is the following process: given an arbitrary partitioning of input variables into m disjoint blocks, a random m-block restriction picks a uniformly random block $\ell \in [m]$ and fixes all variable outside the chosen block ℓ to 0 or 1 according to some distribution. Note that we can use a random block restriction to simulate

the first two types of (pseudo-)random restrictions above. For example, to simulate a truly r-random block restriction, we first randomly partition the variables into $m = 1/r$ disjoint blocks, where each variable is assigned to block $i \in [m]$, independently, with probability $1/m$. Then we apply an m-random block restriction based on the partition in the previous step, by fixing the variables outside the selected block uniformly at random. Similarly, to simulate a pseudorandom restriction using limited-wise independence, we can partition (hash) the variables into disjoint blocks limited-wise independently, and apply a random block restriction where we fix the variables also using a limited-wise independent distribution.

2.3 Useful tools for analyzing PTFs

▶ **Definition 6** (δ-concentrated PTFs)**.** Let $p\colon \{0,1\}^n \to \mathbb{R}$ be a degree-Δ multi-linear polynomial and $f = \operatorname{sgn}(p)$. For parameters $0 < \delta \leq 1/2$ and $\lambda \geq 1$, we call p (and f) (δ, λ)-*concentrated* if $\mathbf{Var}[p] \leq (162 \cdot \log(1/\delta))^{-\lambda \cdot \Delta} \cdot \mathbf{Exp}\left[p^2\right]$, where $\mathbf{Exp}$ and $\mathbf{Var}$ denote the *expectation* and the *variance*, respectively, under the uniform distribution over $\{0,1\}^n$. We refer to $(\delta, 1)$-concentrated polynomials as δ-concentrated.

A useful property of concentrated PTFs is that they are close to an explicit constant.

▶ **Lemma 7** (Concentrated implies close to constant)**.** *For any $0 < \delta \leq 1/2$, if a PTF $f = \operatorname{sgn}(p)$ is δ-concentrated, then f is δ-close to the constant function* $\operatorname{sgn}(\mathbf{Exp}[p])$.

For a multi-linear polynomial $p\colon \{0,1\}^n \to \mathbb{R}$, it easy to see that the constant function $\operatorname{sgn}(\mathbf{Exp}[p])$ from Lemma 7 is efficiently computable for a given polynomial p.

The following is a random restriction lemma for PTFs which says that a low-degree PTF is likely to become concentrated under a (truly) random block restriction.

▶ **Lemma 8** (Random block restriction lemma [11])**.** *For any $0 < \delta < 1$ and any positive integers m, λ, let $\mathcal{B}_m$ be a m-block random restriction that fixes variables uniformly at random. Then for degree-Δ PTF f whose variables are partitioned into m blocks, we have*

$$\mathbf{Pr}_{\rho \sim \mathcal{B}_m}\left[f_\rho \text{ is not } (\delta, \lambda)\text{-concentrated}\right] \leq m^{-1/2} \cdot (\log m \cdot \log(1/\delta))^{O\left(\lambda \cdot \Delta^2\right)}.$$

There is also a derandomized version of the above random block restriction lemma.

▶ **Lemma 9** (Pseudorandom block restriction lemma [11])**.** *For any $0 < \delta, \gamma < 1$ and any positive integers m, λ, there is a polynomial-time algorithm for sampling a m-block random restriction $\mathcal{B}'_m$, that uses at most $m^\gamma \cdot \log n$ random bits, so that the following holds. For any n-variate degree-Δ PTF f whose variables are partitioned into m blocks, we have*

$$\mathbf{Pr}_{\rho \sim \mathcal{B}'_m}\left[f_\rho \text{ is not } (\delta, \lambda)\text{-concentrated}\right] \leq m^{-1/2} \cdot (\log m \cdot \log(1/\delta))^{O\left(\lambda \cdot \Delta^2/\gamma\right)}.$$

Moreover, $\mathcal{B}'_m$ fixes the variables $(192 \cdot \Delta \cdot \log(1/\delta))$-wise independently.

3 #SAT algorithm for PTF circuits

To get our Circuit-SAT algorithm for circuits with sparse PTF gates, we generalize the analysis of the Circuit-SAT algorithm for small LTF circuits in [5]. An oversimplified description is as follows. We show that for a depth-d circuit with a slightly super-linear number of wires, whose gates are sparse PTFs, there exists a shallow decision tree such that, for most of the leaves, the circuit restricted to that leaf can be "approximated" by some depth-$(d-1)$ circuit. Then we recursively apply a Circuit-SAT algorithm to depth-$(d-1)$ circuits. However, to actually implement this idea, we need three ingredients, which we describe in detail below.

Satisfiability for conjunctions of sparse PTFs. First, we need a base-case algorithm. In the case of LTF circuits in [5], the base case is a conjunction of LTFs, and there is a known algorithm by Williams [28] for such circuits. In contrast, in our case, the base case is a conjunction of sparse PTFs. Using the polynomial method in circuit complexity, we are able to design a Circuit-SAT algorithm for such circuits. More specifically, the algorithm is based on the framework for designing satisfiability algorithms developed by Williams [27, 28]. The idea is to transform a given constant-depth circuit into a low-degree probabilistic polynomial and solve satisfiability by evaluating the polynomial on all points in a faster-than-brute-force manner. Applying this idea naively, we get a randomized SAT algorithm that makes error. Such a base-case algorithm would result in the final SAT algorithm for PTF circuits that also makes error. However, using some derandomization ideas similar to those in [3, 21], we are able to obtain a *deterministic* base-case algorithm that can count the number of satisfying assignments. This allows us to make our final SAT algorithm for PTF circuits to be *zero-error* randomized algorithm that *counts* the number of satisfying assignments.

▶ **Lemma 10.** *There exists a deterministic algorithm that counts the number of satisfying assignments of every n-variate circuit C that is a conjunction of k s-sparse PTF gates, where $s \geq n$, such that the algorithm runs in time at most $2^{n-\left(\frac{n}{\sqrt{s}\cdot(\log s)^{O(1)}\cdot\log^2 k}\right)^{1/2}}$.*

Depth reduction for sparse PTF circuits with few wires. Secondly, to construct the aforementioned decision tree, we need a random restriction lemma showing that, under a (truly) random restriction, a gate in the circuit is likely to be close to constant. More specifically, Chen et al. [5] showed that using such a random restriction lemma, one can get a shallow decision tree such that, restricted to most of the leaves, a circuit with few wires will have many of its bottom-layer gates becoming close to constant, so that we can replace them with actual constants, and the depth decreases by one if we futher remove the rest of the few bottom-layer gates that are not close to constant. In our case, we need a similar restriction lemma for sparse PTFs. It is easy to see that a sparse PTF is likely to become a low-degree PTF under a mild random restriction. Combining this observation with the block restriction lemma for low-degree PTFs (Lemma 8), it is not difficult to show that for any $\left(s \leq n^{O(1)}\right)$-sparse PTF f, $\mathbf{Pr}_{\rho\sim\mathcal{R}_r}[f_\rho \text{ is not } \delta\text{-close to an explicit constant}] \leq r^{\Omega(1)}$, for some very small δ, where $\mathcal{R}_r$ denotes the truly r-random restriction. Using this structual result for sparse PTFs and the idea in [5] (see Section 4.1.1 of [5]), we get the following.

▶ **Lemma 11.** *For any integer $d \geq 2$ and any $(\log n)^{-1} \ll \varepsilon < 1$, let*

- *$\beta = E \cdot \varepsilon$ where E is some constant, $s \leq n^{O(1)}$, $\delta = \exp\left(-n^{\Omega(\beta^3)}\right)$, and*
- *C be any depth-d, n-variate, s-sparse PTF circuit with at most $w = n^{1+\varepsilon}$ wires.*

Then there exists a decision tree T of depth $n - n^{1-\beta}$ such that, for a random leaf σ of T, with probability at least $1 - \exp(-n^{\varepsilon})$, we have the following: C_σ is a depth-d circuit of wire complexity at most w such that its bottom layer has at most n gates that are δ-close to an explicit constant and at most n^β gates that are not δ-close to an explicit constant. Moreover, such a tree can be constructed in zero-error randomized time $\tilde{O}\left(2^{n-n^{1-\beta}}\right)$.

Enumerating minority outputs of sparse PTFs. Finally, in our main algorithm, we will need to apply the depth reduction lemma (Lemma 11) to the circuit to conclude that many of the gates at the bottom layer will become close to constant so that we can replace them with actual constants. This changes the function of the circuit and we need to deal with the inputs where these gates do not evaluate to their majority values. This issue can be handled

if given a sparse PTF we can find the set of all inputs where it evaluates to its minority value, in a relatively efficient way. As shown in [5], there is an efficient way to do this for functions whose satisfiability can be decided in polynomial time, such as LTFs. However, we cannot apply this for sparse PTFs since there is no known polynomial-time SAT algorithm for sparse PTFs. We overcome this issue for sparse PTFs by reducing to the case of LTFs, using the following observation from Chen and Santhanam [4], which says that for a collection of sub-quadratic many monomials, there exists a decision tree of not-too-many leaves such that, under each leaf, each of the monomials becomes a single literal. As a result, we get the following way to enumerate the set of minority-value inputs for a sub-quadratically sparse PTFs in non-trivial time. (See Section A for the proof).

▶ **Lemma 12.** *Let $f\colon \{0,1\}^n \to \{0,1\}$ be a s-sparse PTF with coefficients of bit complexity $\mathsf{poly}(n)$, where $s \geq n$ and let S be the set of inputs on which f evaluates to 0 (or 1). Then S can be enumerated in time $\left(2^{n-\Omega(n^2/s)} + |S|\right) \cdot \mathsf{poly}(n)$.*

Putting it all together. Let $\varepsilon_d = 1/\left(E^{3^{d-1}}\right)$ and $\beta_d = E \cdot \varepsilon_d$, where E is a sufficiently large constant. We can show the following.

▶ **Theorem 13.** *For any integer $d \geq 1$, the number of satisfying assignments of a depth-d, n-variate circuit with $(n^{2-10\beta_2})$-sparse PTF gates and at most $n^{(1+\varepsilon_d)}$ wires can be computed by a zero-error randomized algorithm in time $\mathsf{poly}(n) \cdot 2^{n-n^{\Omega\left(\beta_d^3\right)}}$.*

▶ **Definition 14** (Skew Circuits)**.** We say that a circuit C is (d,n,t,s)-skew if it is a n-variate circuit that can be expressed as a conjunction of some circuit C' and at most t s-sparse PTFs, where C' is a depth-d circuit with s-sparse PTF gates and has at most $w = n^{1+\varepsilon_d}$ wires. We call C' the skew subscircuit of C.

Let $\mathcal{T}(d,n,t,s)$ denote the supremum, over all (d,n,t,s)-skew circuits C, of the randomized running time of counting the number of satisfying assignments of C.

Using the depth reduction lemma (Lemma 11) and the enumeration lemma (Lemma 12), we can obtain the following lemma which says that we can reduce the task of counting satisfying assignments of depth-d circuits to that of depth-$(d-1)$ circuits.

▶ **Lemma 15.** *If $s \leq n^{2-5\beta_d}$, then*

$$\mathcal{T}(d,n,t,s) \leq 2^{n-n^{1-2\beta_d}} \cdot 2^{n^{\beta_d}} \cdot \mathcal{T}\left(d-1, n^{1-2\beta_d}, t+2n, s\right) + \mathsf{poly}(n) \cdot 2^{n-n^{\Omega\left(\beta_d^3\right)}}.$$

The proof of the above lemma is similar to that in [5]. We give a detailed proof in Sectoin B.

Given the recursion in Lemma 15, it is not difficult to prove Theorem 13 by solving the recursion and using the SAT algorithm for conjunctions of sparse PTFs (Lemma 10) as the base case. We refer the reader to the full version for the proof.

4 Quantified derandomization for PTF circuits

At a high level, our quantified derandomization algorithm follows the approach of [8]. Given a circuit C with at most B minority-value inputs, we find a restriction ρ such that ρ leaves a large number, say n', of variables unrestricted, and that C_ρ is very close to some simple function $\tilde{C}$ (say they agree on all but at most $1/6$ fraction of inputs). Then the number of minority-value inputs for $\tilde{C}$ is at most $B + 2^{n'}/6$. If B is also at most $2^{n'}/6$, then we can determine the required majority value for C by finding the majority value $\tilde{C}$, which is a

simple function. This approach is also used by Tell [24] to get a quantified derandomization algorithm for LTF circuits with a slightly super-linear number of wires.

Let's first consider a depth-2 LTF circuit with few wires. In [5], Chen, Santhanam and Srinivasan proved a random restriction lemma for LTFs, which says that under a random restriction, an LTF is likely to become very close to an explicit constant. Using this result, one gets that under such a random restriction, many of the gates in the bottom layer of the circuit are expected to become close to constants. Since the circuit has only a few wires, one can further fix a small number of variables so that only those gates that are close to constants are left. Finally, by replacing these gate with their majority values, we obtain a single LTF that is close to the original depth-2 circuit.

Such a random restriction lemma was extended to low-degree PTFs in [11], so we can conclude the same for low-degree PTF circuits. One important issue, though, is that the above "depth reduction" argument only holds for *random* restrictions (but with high probability). So to get quantified derandomization, one will need to consider all possible restrictions. To handle this issue, Tell [24] derandomized the random restriction lemma for LTFs mentioned above so that such a restriction can be sampled using few random bits. As a result, one only needs to consider a much smaller sample space of restrictions.

Pseudorandom restrictions for PTFs. The pseudorandom restriction lemma for LTFs in [24] is obtained using a PRG for LTFs. One way to extend it to PTFs is to use a PRG for PTFs. However, unlike LTFs, for which a PRG with a very short seed is known, all known PRGs for PTFs have a large seed length (for small error, which is needed for the argument). In fact, the only PRG that we can use in this case is the one in Corollary 5, and it would give a quantified derandomization algorithm running in time $\exp\left(\exp\left(\sqrt{\Delta \cdot \log n}\right)\right)$. To get quasi-polynomial running time, we use a powerful pseudorandom block restriction lemma for PTFs (Lemma 9), which needs only a poly-logarithmic number of random bits, and convert it into the following pseudorandom restriction lemma. (The proof is in Section C.)

▶ **Lemma 16** (Pseudorandom restriction lemma for low-degree PTFs). *For any constant $c > 0$, any $\alpha < 1$ and any positive integer Δ such that $\Delta \ll \sqrt{\alpha \cdot \log n / \log\log n}$, there is a random restriction $\mathcal{R}$ such that the following holds:*

- *$\mathcal{R}$ picks the unrestricted variables $(\log n)$-wise independently, each with probability $n^{-\alpha}$.*
- *$\mathcal{R}$ fixes the variables $(600 \cdot c \cdot \Delta \cdot \log n)$-wise independently.*
- *$\mathcal{R}$ can be sampled in polynomial time using $(\log n)^{O(\Delta^2)}$ random bits.*
- *For any degree-Δ PTF f on n variables, $\mathbf{Pr}_{\rho\sim\mathcal{R}}[f_\rho$ is not $(n^{-c}, 3)$-concentrated$] \leq n^{-\alpha/3}$.*

Bias preservation for PTFs. Now we need to apply the above idea to a depth-d circuit C. It seems that all we need to do is applying the pseudorandom restriction $d-1$ times. While this is true, the analysis is much more subtle. For example, after applying the first pseudorandom restriction ρ_1, we get a new circuit $\tilde{C}$ of depth $(d-1)$ on some n' variables so that it agrees with C_{ρ_1} on all but at most say $2^{n'}/6$ inputs. Now consider a subsequent restrictions ρ'. Note that the final number of unrestricted variable n'' after ρ' is much smaller than n'. Therefore, $(C_{\rho_1})_{\rho'}$ and $\tilde{C}_{\rho'}$ can disagree on all the inputs (since $2^{n'}/6 \gg 2^{n''}$) so $\tilde{C}_{\rho'}$ cannot be used to determine the correct output of $(C_{\rho_1})_{\rho'}$, which is also the correct output of C. This issue can be handled if those bottom layer gates that become close to constant after applying one step of pseudorandom restriction will remain close to the *same* constant for subsequent pseudorandom restrictions. Such a "bias preservation lemma" for LTFs is also proved in [24], again using a PRG for LTFs. For PTFs, we use an observation in [11], which

says that a concentrated PTF is likely to remain concentrated under any random restriction that fixes variables limited-wise independently.

▶ **Lemma 17** (see Lemma 4.2 and Claim 7.7 of [11]). *Let $f = \mathrm{sgn}(p)$ be any degree-Δ PTF that is $(\delta, \lambda+1)$-concentrated. Let ρ be a random restriction that fixes any subset of variables according to some $(192 \cdot \Delta \cdot \log(1/\delta))$-wise independent distribution. Then with probability at least $1-\delta$ we have: (1) f_ρ is (δ, λ)-concentrated, and (2) $\mathrm{sgn}(\mathbf{Exp}(p_\rho)) = \mathrm{sgn}(\mathbf{Exp}(p))$.*

The above lemma means that if a PTF is $(\delta, 2)$-concentrated and hence close to some constant. Then the restricted PTF is likely to remain close to the same constant.

Quantified derandomization for low-degree PTF circuits. Let $\mathcal{G}$ be a class of Boolean functions, we say that a circuit C is a $(n, d, w, \Delta, \mathcal{G})$-low-degree PTF circuits if: (1) C is an n-variate circuit of depth-d with at most w wires, and (2) C has degree-Δ PTFs as its gates except for the top gate, which is a function from $\mathcal{G}$.

For a class of Boolean functions $\mathcal{G}$, we denote by $\mathrm{Appr}_{n,\epsilon}(\mathcal{G})$ the running time, given an n-variate function g from $\mathcal{G}$, of approximating the acceptance probability of g to within an additive error ϵ. We can show the following.

▶ **Theorem 18.** *For any constant $E \geq 11$ and any positive integers Δ and d such that $\Delta \ll \sqrt{\varepsilon_d \cdot \log n / \log\log n}$, where $\varepsilon_d = E^{-2(d-1)}$, let $\mathcal{C}$ be the class of $\left(n, d, n^{1+\varepsilon_d}, \Delta, \mathcal{G}\right)$-low-degree PTF circuits. Then the $\left(\mathcal{C}, 2^{n^{1-7/E}}\right)$-quantified derandomization problem can be solved in time $2^{(\log n)^{O\left(\Delta^2\right)}} \cdot \mathrm{Appr}_{n,1/6}(\mathcal{G})$.*

Theorem 18 implies Theorem 3 for low-degree PTF circuits since we can always add a dummy gate (e.g., AND) to the top of a PTF circuits; this only increase the depth by 1.

Theorem 18 is obtained by iteratively applying the pseudorandom restriction lemma (Lemma 16) to reduce the depth of the circuit until the circuit has depth 1. The following lemma shows how to do this in one step.

▶ **Lemma 19.** *For any constants $E \geq 11$, $c > 0$, any $\varepsilon \leq 1/(7E)$, and any positive integer Δ such that $\Delta \ll \sqrt{E \cdot \varepsilon \cdot \log n / \log\log n}$, there is a polynomial time algorithm that, given a $\left(n, d, n^{1+\varepsilon}, \Delta, \mathcal{G}\right)$-low-degree PTF circuit C and a random seed of length $(\log n)^{O\left(\Delta^2\right)}$, outputs the following with probability at least $1 - n^{\varepsilon}$:*

- *A restriction $\rho \in \{0, 1, *\}^n$ that leaves $n' = n^{1-3E\cdot\varepsilon}$ variables unrestricted and that the restricted variables are fixed $(600 \cdot c \cdot \Delta \cdot \log n)$-wise independently.*
- *A $\left(n', d-1, (n')^{1+7E\cdot\varepsilon}, \Delta, \mathcal{G}\right)$-sparse PTF circuit $\tilde{C}$ such that for all subsequent random restriction ρ' that fixes the variables in a $(600 \cdot c \cdot \Delta \cdot \log n)$-wise independent manner, with probability $1 - n^{-c}$ over ρ', it holds that $\tilde{C}_{\rho'}$ is n^{-c}-close to $(C_\rho)_{\rho'}$.*

The proof of Lemma 19 is similar to that in [24], which is based on the argument in [5], but requires some critical modifications. A sketched proof is given in Section D. For the proof of Theorem 18, we refer the reader to the full version.

5 PRG for PTF circuits

In this section, we give a high-level description of our PRG for small PTF circuits. The detailed proof is presented in Section E.

Our PRG is based on the Nisan-Wigderson generator (NW PRG) [19]. To fool a class $\mathcal{C}$ of Boolean functions f, the NW PRG construction requires a "hard function" h that cannot be computed correctly on significantly more than a half of all possible inputs by any Boolean

function g in a related class $\tilde{\mathcal{C}}$ of "slightly more powerful" functions than those from $\mathcal{C}$. Thus, sufficiently strong average-case lower bounds against the class $\tilde{\mathcal{C}}$ can be used to build a PRG fooling the class $\mathcal{C}$. In our case, the class $\mathcal{C}$ contains all those n-variate Boolean functions that are computable by constant depth-d circuits with at most $s \ll n$ PTF gates of degree-Δ. Our main observation is that the corresponding class $\tilde{\mathcal{C}}$ (for which we require average-case lower bounds) is the class of Boolean functions computable by constant depth-d circuits with at most s PTF gates of degree $\Delta' = \alpha \cdot \Delta$, for some parameter $\alpha \geq 1$ that we can control (and which will determine the seed size of our PRG). That is, the class $\tilde{\mathcal{C}}$ is the same as $\mathcal{C}$, except for a somewhat higher degree Δ' of the allowed PTF gates.

To illustrate the idea of our analysis of the NW PRG for PTF circuits, we consider the special case of a single n-variate PTF f of degree Δ. That is, $f = \mathrm{sgn}(p(x_1, \ldots, x_n))$ for some degree-Δ multi-linear polynomial $p\colon \{0,1\}^n \to \mathbb{R}$. Suppose that the NW generator based on some "hard" Boolean function h failed to ϵ-fool this PTF f. First, the standard NW analysis shows that the function $h(z)$ can be computed, with probability at least $1/2 + \epsilon/n$, by (possibly the negation of) the function

$$g(z) = f(h_1(z), h_2(z), \ldots, h_i(z), b_{i+1}, \ldots, b_n), \tag{1}$$

for some $1 \leq i \leq n$, fixed bits $b_{i+1}, \ldots, b_n$, and Boolean functions $h_1, \ldots, h_i$, where each $h_j(z)$ depends on at most some α bits in z, for a parameter $a \geq 1$ coming from the NW construction (the maximum overlap between pairs of sets in the NW design; see Section E for details). It is well known that every Boolean function on α inputs can be written as a multi-linear polynomial of degree α over the reals. Plugging in these polynomials for the function h_j's in Equation (1), we get that $g(z)$ is a PTF of degree at most $\Delta' = \alpha \cdot \Delta$. Hence, to ensure that this NW generator based on h is indeed ϵ-fooling for degree-Δ PTFs, we just need h to be such that no PTF of degree-$(\alpha \cdot \Delta)$ can compute $h(z)$ on more than $1/2 + \epsilon/n$ of inputs z. Such hard functions h turn out to be easy to construct. For example, we use the average-case hard function for low-degree PTF circuits due to Nisan [18].

The parameters of our PRG $G\colon \{0,1\}^r \to \{0,1\}^n$ (its error ϵ and seed length r) depend on the strength of the average-case lower bound for the hard function h. To get a short seed r, one needs to maximize the aforementioned parameter α, ideally setting $\alpha = \log n$ (as is the case for a standard application of the NW construction). However, we also need to prove (average-case) lower bounds against PTFs of degree $\alpha \cdot \Delta$, where virtually nothing is known for the degree $\log n$. Thus we are forced to set $\alpha \ll \log n$, which limits the stretch of our PRG to be at most only super-polynomial. On the other hand, for such a small α, our hard function h has exponentially small correlation with degree-$(\alpha \cdot \Delta)$ PTFs, thereby allowing our PRG to have an exponentially small error ϵ.

6 Open problems

An important open problem is to get a nontrivial Circuit-SAT algorithm for circuits with degree-2 PTF gates. Our algorithm only works for the case where the PTF gates have a sub-quadratic number of monomials, so it does not work for the degree-2 case in general. Such an algorithm is not known even for a single degree-2 PTF. Another interesting open problem is to derandomize our zero-error randomized algorithms to get deterministic #Circuit-SAT algorithms of similar time complexity. Can we get any nontrivial standard derandomization for constant-depth PTF (LTF) circuits of small wire complexity? For PRGs, can we get a nontrivial PRG for depth-2 LTF circuits with a super-linear number of gates?

References

1 Josh Alman, Timothy M. Chan, and R. Ryan Williams. Polynomial representations of threshold functions and algorithmic applications. In *FOCS*, pages 467–476, 2016.

2 Martin Anthony. *Discrete Mathematics of Neural Networks: Selected Topics*. SIAM monographs on discrete mathematics and applications. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2001. doi:10.1137/1.9780898718539.

3 Timothy M. Chan and Ryan Williams. Deterministic APSP, orthogonal vectors, and more: Quickly derandomizing razborov-smolensky. In *SODA*, pages 1246–1255, 2016.

4 Ruiwen Chen and Rahul Santhanam. Improved algorithms for sparse MAX-SAT and MAX-k-CSP. In *SAT*, pages 33–45, 2015.

5 Ruiwen Chen, Rahul Santhanam, and Srikanth Srinivasan. Average-case lower bounds and satisfiability algorithms for small threshold circuits. In *CCC*, pages 1:1–1:35, 2016.

6 Shiteng Chen and Periklis A. Papakonstantinou. Depth-reduction for composites. In *FOCS*, pages 99–108, 2016.

7 Mikael Goldmann, Johan Håstad, and Alexander A. Razborov. Majority gates vs. general weighted threshold gates. *Computational Complexity*, 2:277–300, 1992.

8 Oded Goldreich and Avi Wigderson. On derandomizing algorithms that err extremely rarely. In *STOC*, pages 109–118, 2014.

9 Johan Håstad. Almost optimal lower bounds for small depth circuits. In S. Micali, editor, *Randomness and Computation*, pages 143–170, Greenwich, Connecticut, 1989. Advances in Computing Research, vol. 5, JAI Press.

10 Russell Impagliazzo, Ramamohan Paturi, and Stefan Schneider. A satisfiability algorithm for sparse depth two threshold circuits. In *FOCS*, pages 479–488, 2013.

11 Valentine Kabanets, Daniel M. Kane, and Zhenjian Lu. A polynomial restriction lemma with applications. In *STOC*, pages 615–628, 2017.

12 Daniel Kane and Sankeerth Rao. A PRG for Boolean PTF of degree 2 with seed length subpolynomial in ϵ and logarithmic in n. In *CCC*, 2018.

13 Daniel M. Kane. A structure theorem for poorly anticoncentrated gaussian chaoses and applications to the study of polynomial threshold functions. In *FOCS*, pages 91–100, 2012.

14 Shachar Lovett and Srikanth Srinivasan. Correlation bounds for poly-size AC^0 circuits with $n^{1-o(1)}$ symmetric gates. In *APPROX/RANDOM*, pages 640–651, 2011.

15 Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5(4):115–133, 1943.

16 Raghu Meka and David Zuckerman. Pseudorandom generators for polynomial threshold functions. *SIAM J. Comput.*, 42(3):1275–1301, 2013.

17 Saburo Muroga, Iwao Toda, and Satoru Takasu. Theory of majority decision elements. *Journal of the Franklin Institute*, 271:376–418, 1961.

18 Noam Nisan. The communication complexity of threshold gates. In *Proceedings of Combinatorics, Paul Erdős is Eighty*, pages 301–315, 1994.

19 Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994.

20 Takayuki Sakai, Kazuhisa Seto, Suguru Tamaki, and Junichi Teruyama. Bounded depth circuits with weighted symmetric gates: Satisfiability, lower bounds and compression. In *MFCS*, pages 82:1–82:16, 2016.

21 Suguru Tamaki. A satisfiability algorithm for depth two circuits with a sub-quadratic number of symmetric and threshold gates. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:100, 2016.

22 Roei Tell. Improved bounds for quantified derandomization of constant-depth circuits and polynomials. In *CCC*, pages 13:1–13:48, 2017.

23 Roei Tell. A note on the limitations of two black-box techniques in quantified derandomization. *Electronic Colloquium on Computational Complexity (ECCC)*, 24:187, 2017.

24 Roei Tell. Quantified derandomization of linear threshold circuits. In *STOC*, 2018.

25 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1-3):1–336, 2012.

26 Ryan Williams. Improving exhaustive search implies superpolynomial lower bounds. In *STOC*, pages 231–240, 2010.

27 Ryan Williams. Non-uniform ACC circuit lower bounds. In *CCC*, pages 115–125, 2011.

28 Ryan Williams. New algorithms and lower bounds for circuits with linear threshold gates. In *STOC*, pages 194–202, 2014.

A Enumerating minority-value inputs: proof of Lemma 12

We first need the folowing.

▶ **Proposition 20** (see Section 4.1 of [4]). *Let $\phi_1, \ldots, \phi_s$ be a sequence of terms whose literals are from a set of n variables, where $s \geq n$. There exists a decision tree with at most $2^{n-\Omega(n^2/s)}$ leaves such that restricted to each leaf of the tree, ϕ_i contains at most 1 literal, for all $i \in [s]$.*

Proof of Lemma 12. We view f as an LTF of s AND gates. By Proposition 20, there exists a decision tree for f with at most $2^{n-\Omega(n^2/s)}$ leaves such that Φ restricted to each leaf is an LTF. We then go through each leaf σ and enumerate the set of inputs on which f_σ evaluates to 0. Let S_σ be the size of such set. For an LTF, this enumeration takes time $S_\sigma \cdot \mathsf{poly}(n)$ (see, e.g., Proposition 5.2 of [5]). The total running time is the time for going through the leaves of the decision tree, which is at most $2^{n-\Omega(n^2/s)}$, and the time to enumerate the set of inputs evaluating to 0, which is at most $\sum_\sigma S_\sigma \cdot \mathsf{poly}(n) \leq |S| \cdot \mathsf{poly}(n)$. ◀

B Recursion for depth-d circuits: proof of Lemma 15

Let C be any (d, n, t, s)-skew circuit, where its skew subcircuit is C'. To count the number of satisfying assignments of C. We first apply Lemma 11 to C' to get a decision tree with the claimed property. We then count the number of satisfying assignments at each leaves. For those "bad" leaves for which the conditions in Lemma 11 are not satisfied, we will simply do brute force on all $n^{1-2\beta_d}$ variables. The time to perform this is

$$2^{n-n^{1-2\beta_d}} \cdot \exp(-n^{\varepsilon_d}) \cdot 2^{n^{1-2\beta_d}} \leq 2^{n-n^{\varepsilon_d}}. \tag{2}$$

Next, consider a "good" leaf σ that satisfies the conditions in Lemma 11. We now describe how to count the number of satisfying assignments of C_σ. We call a gate *imbalanced* if it is δ-close to an explicit constant and *balanced* otherwise. Let $(g_1, \ldots, g_{\ell \leq n})$ be the set of imbalanced gates and $(a_1, \ldots, a_\ell)$ be their majority values. Let $(h_1, \ldots, h_{t \leq n^{\beta_d}})$ be the set of balanced gates.

We first count the number of satisfying assignments of C_σ in the following subset of inputs $S = \{x : \exists\, i \in [\ell] \text{ for which } g_i(x) \neq a_i\}$. To do so, for each of the imbalanced gates, we enumerate the set of inputs on which it evaluates to its minority value, and keep those that satisfy the circuit C_σ. By Lemma 12, the running time for this is

$$\mathsf{poly}(n) \cdot \left(2^{n^{1-2\beta_d} - \Omega\left(\frac{n^{2\cdot(1-2\beta_d)}}{s}\right)} + 2^{n^{1-2\beta_d}} \cdot \delta \right), \tag{3}$$

where $\delta = \exp\left(-n^{\Omega\left(\beta_d^3\right)}\right)$. Note that for $s \leq n^{2-5\beta_d}$, we have

$$2^{n^{1-2\beta_d}-\Omega\left(\frac{n^{2\cdot(1-2\beta_d)}}{s}\right)} \leq 2^{n^{1-2\beta_d}-\Omega\left(n^{\beta_d}\right)} \leq 2^{n-n^{\Omega\left(\beta_d^3\right)}},$$

so Equation (3) is at most

$$\mathsf{poly}(n) \cdot 2^{n^{1-2\beta_d}-n^{\Omega\left(\beta_d^3\right)}}. \tag{4}$$

We do this for every imbalanced gate and obtain a set of satisfying inputs. In the end we simply take the union of these sets to get the satisfying assignments in S.

Next, we counts the number of satisfying assignments in $T = \{0,1\}^n - S$. Let $C'_{\sigma,a}$ be the circuit with those imbalanced gates in C'_σ replaced with their majority values (i.e., the values given by $(a_1, \ldots, a_\ell)$). Instead of counting the number of satisfying assignments for the original circuit C_σ, we consider the following circuit:

$$D = C'_{\sigma,a} \wedge \bigwedge_{i\colon a_i=-1} g_i \wedge \bigwedge_{i\colon a_i=1} \neg g_i.$$

It is easy to see that $D(x) = 0$ for every $x \in S$ and $D(x) = C_\sigma(x)$ for every $x \in T$. We now need to count the number of satisfying assignments of D. We first partition T into 2^t subsets, each of which is indexed by some $b = (b_1, \ldots, b_t) \in \{0,1\}^t$, where the subset T_b given by the index b is

$$T_b = \{x : x \in T,\, h_1(x) = b_1, \ldots,\, h_t(x) = b_t\}.$$

To count the number of satisfying assignments of D in T_b. We consider the following circuit:

$$E_b = D_b \wedge \bigwedge_{i\colon b_i=-1} h_i \wedge \bigwedge_{i\colon b_i=1} \neg h_i,$$

where D_b is the circuit D with the balanced gates replaced by the values $b_1, \ldots, b_t \in \{0,1\}$. Again, we have $E_b(x) = 0$ for every $x \in [n] - T_b$ and $E_b(x) = D(x)$ for every $x \in T_b$. Now our task is reduced to counting the number of satisfying assignments of E_b for each $b \in \{0,1\}^t$. But note that each E_b is a conjunction of some depth-$(d-1)$ circuit (i.e., the skew subcircuit of E_b) and k s-sparse PTFs, where $k = t + n + n^\beta \leq t + 2n$. Also, the skew subcircuit has at most $n^{1+\varepsilon_d}$ wires, and we have

$$n^{1+\varepsilon_d} \leq \left(n^{1-2\beta_d}\right)^{1+\varepsilon_{d-1}}.$$

Therefore, each E_b is a $\left(d-1, n^{1-2\beta_d}, t+2n, s\right)$-skew circuits, and its number of satisfying assignments can be computed in time $\mathcal{T}\left(d-1, n^{1-2\beta_d}, t+2n, s\right)$. Then the total time for counting the number of satisfying assignments of the original circuit C_σ in the subset T is

$$2^t \cdot \mathcal{T}\left(d-1, n^{1-2\beta_d}, t+2n, s\right) \leq 2^{n^{\beta_d}} \cdot \mathcal{T}\left(d-1, n^{1-2\beta_d}, t+2n, s\right). \tag{5}$$

Therefore, by Equation (4) and Equation (5), counting the number of satisfying assignments of C_σ can be done in time

$$\mathsf{poly}(n) \cdot 2^{n^{1-2\beta_d}-n^{\Omega\left(\beta_d^3\right)}} + 2^{n^{\beta_d}} \cdot \mathcal{T}\left(d-1, n^{1-2\beta_d}, t+2n, s\right). \tag{6}$$

There are at most $L = 2^{n-n^{1-2\beta_d}}$ such leaves. Multiplying L by the running time in Equation (6) and combining Equation (2) yields the desired running time.

C Pseudorandom restriction lemma for PTFs: proof of Lemma 16

We define $\mathcal{R}$ by describing the following process of sampling a random restriction from $\mathcal{R}$:

1. Hash the variables into n^α blocks $(\log n)$-wise independently.
2. Apply a $\left(n^{\alpha/2}\right)$-block pseudorandom restriction from Lemma 9 for degree-Δ PTFs, with parameters
   - $\delta = n^{-c}$ and $\lambda = 3$.
   - $\gamma = (c_1 \cdot \Delta^2 \cdot \log\log n)/(\alpha \cdot \log n)$, where c_1 is a sufficiently large constant (note that $\gamma < 1$ for $\Delta \ll \sqrt{\alpha \cdot \log n / \log\log n}$).

We now argue that the random restriction $\mathcal{R}$ has the desired properties. For the first item, it is easy to see from the above that $\mathcal{R}$ picks the set of unrestricted variables $(\log n)$-wise independently, each with probability $1/n^{\alpha/2}$. The second item follows from Lemma 9 that the pseudorandom block restriction fixes the variables $(600 \cdot c \cdot \Delta \cdot \log n)$-wise independently. For the third item, note that to sample from $\mathcal{R}$, we need $\mathsf{polylog}(n)$ random bits for its first step, and the number of random bits for its second step is

$$n^{\alpha\cdot\gamma} \cdot \log n \leq (\log n)^{O(\Delta^2)}.$$

Finally, for the last item, note that in the above process of sampling $\mathcal{R}$, for any partition into n^α blocks generated in the first step, by Lemma 9, the probability over the restrictions in the second step that the restricted PTF is not $(n^{-c}, 3)$-concentrated is at most $n^{-\alpha/2} \cdot (\log n)^{O(\Delta^2/\gamma)}$. Thus,

$$\mathbf{Pr}_\rho[f_\rho \text{ is not } (n^{-c}, 3)\text{-concentrated}] \leq n^{-\alpha/2} \cdot (\log n)^{O(\Delta^2/\gamma)} \leq n^{-\alpha/2} \cdot n^{\alpha/6} \leq n^{-\alpha/3}.$$

D Depth reduction via pseudorandom restrictions: proof of Lemma 19

Let $\beta = E \cdot \varepsilon$ and $p = n^{-\beta}$. The restriction ρ consists of three sub-restrictions.

ρ_1: Preprocessing. Fix each of the variables with fan-out greater than $2n^\varepsilon$ using a $(600 \cdot c \cdot \Delta \cdot \log n)$-wise independent distribution. Since the number of wires is at most $n^{1+\varepsilon}$, it can be easily seen that the number of variables needed to be fixed is at most $n^{1+\varepsilon}/(2n^\varepsilon) = n/2$.

ρ_2: Pseudorandom restriction to simplify PTFs. Let ρ_2 be a random restriction from Lemma 16 with parameters $\alpha_2 = \beta$ and $c_2 = 2c$. Note that ρ_2 fixes the variables $(600 \cdot c \cdot \Delta \cdot \log n)$-wise independently. Now by Lemma 16, after ρ_2, we expect all but at most a fraction of $n^{-\beta/5}$ of the gates in the bottom layer to become $(n^{-2c}, 3)$-concentrated. Moreover, since the number of unrestricted variables is picked in a $(\log n)$-wise independent manner, by a Chernoff-type concentration bound (for k-wise independence), the fan-in of each of the non-concentrated gates (there are only about a fraction of $n^{-\beta/5}$ of such gates) will shrink by a factor of p with high probability, assuming they have large fan-ins. Then we can expect to eliminate all those non-concentrated gates by fixing a small number of variables. As for the gates with small fan-ins, using a simple graph theoretic argument along with the condition given by the preprocessing step, we can also eliminate those gate by fixing a few variables. More precisely, as in [5, 24], it can be shown that with probability except $O\left(n^{-\beta/10}\right)$, over the random restriction ρ_2, the following holds: there is a set T of variables such that all the bottom layer gates that are not $(n^{-2c}, 3)$-concentrated can be replaced by constants after fixing the variables in T. The number of unrestricted variables after applying ρ_2 and fixing T is at least $n^{1-3E\cdot\varepsilon}$.

ρ_3: Eliminate non-concentrated gates. We will use a $(600 \cdot c \cdot \Delta \cdot \log n)$-wise independent distribution to fix the variable in the set T described above. Note that the number of unrestricted variables is at least $n' = n^{1-3E\cdot\varepsilon}$. We may further fix additional variables so that the number of unrestricted variables is exactly n'. Although this restriction eliminates all non-concentrated gates in the bottom layer, it may also cause some concentrated gates to become non-concentrated. However, by Lemma 17, the probability that each of these gate is not $(n^{-2c}, 2)$-concentrated is at most n^{-2c}. By the union bound, we get with probability all but $n^{-2c} \cdot n^{1+\varepsilon} \leq n^{-c}$, all these gates remain $(n^{-2c}, 2)$-concentrated.

Obtaining $\tilde{C}$. By above, with probability at least $1 - O\left(n^{-\beta/10}\right)$, we have a restriction ρ such that all the bottom layer gates of C_ρ are $(n^{-2c}, 2)$-concentrated and hence close to some associated constants. Let's call these constants V. $\tilde{C}$ is the circuit obtained from C_ρ by replacing those concentrated gates in the bottom with the constants V. Let's argue that $\tilde{C}_{\rho'}$ and $(C_\rho)_{\rho'}$ are n^{-c}-close to each other for any subsequent random restriction ρ' that fixes the variables $(600 \cdot c \cdot \Delta \cdot \log n)$-wise independently. Consider such a subsequent random restriction ρ' and the restricted circuit $(C_\rho)_{\rho'}$. By Lemma 17, with probability except n^{-2c}, the bottom layer gates of $(C_\rho)_{\rho'}$, which are just the bottom layer gates of C_ρ, are still (n^{-2c})-concentrated. Moreover, they are close to the same constants V. Now by replacing these gates in $(C_\rho)_{\rho'}$ with the constants V, we obtain a circuit C'. By a union bound, C' and $(C_\rho)_{\rho'}$ are (n^{-c})-close to each other. On the other hand, consider the circuit $\tilde{C}$, which is obtained by replacing the concentrated gates in the bottom C_ρ with the constant V. Note that $\tilde{C}_{\rho'} = C'$. Thus, $\tilde{C}_{\rho'}$ and $(C_\rho)_{\rho'}$ are n^{-c}-close to each other. Finally, we need to show that $\tilde{C}$ is a $\left(n', d-1, (n')^{1+4E\cdot\varepsilon}, (n')^{\Delta\cdot E\cdot\varepsilon}, \mathcal{G}\right)$-sparse PTF circuit. As for the number of wires in $\tilde{C}$, note that

$$(n')^{1+7E\cdot\varepsilon} = n^{(1-3E\cdot\varepsilon)\cdot(1+7E\cdot\varepsilon)} \geq n^{1+\varepsilon}.$$

Also, we have $(n')^{2\Delta\cdot\varepsilon} = n^{(1-3E\cdot\varepsilon)\cdot 2\Delta\cdot\varepsilon} \geq n^{\Delta\cdot\varepsilon}$.

E PRG for PTF circuits: proof of Theorem 4

In this section, we present our NW-style PRG for low-degree PTF circuits with few gates.

▶ **Theorem 21.** *There exists a constant $E > 0$ such that for any positive integers α, Δ and any degree-Δ PTF circuit C on n variables with at most $s = n^{\frac{1}{\alpha+1}} \cdot \left(E \cdot 5^{\alpha\cdot\Delta} \cdot \log^2(n) \cdot \log(n/\epsilon)\right)^{-1}$ gates, there exists a* $\mathsf{poly}(n)$*-time computable PRG $G\colon \{0,1\}^r \to \{0,1\}^n$ ϵ-fooling C, with the seed length $r = n^{2/(\alpha+1)}$.*

We first need a (average-case) hard function for such circuits.

▶ **Theorem 22** ([18])**.** *There exists a constant $E > 0$ such that for any degree $\Delta \geq 1$, there exists a polynomial-time computable function $f\colon \{0,1\}^n \to \{0,1\}$ such that for any error parameter ϵ and any n-variate degree-Δ PTF circuit C with at most $n \cdot \left(E \cdot 5^{\Delta} \cdot \log^2(n) \cdot \log(1/\epsilon)\right)^{-1}$ gates, we have*

$$\mathbf{Pr}_{x\sim\{0,1\}^n}\left[C(x) = f(x)\right] \leq \frac{1}{2} + \epsilon.$$

Next we apply the Nisan-Wigderson construction to the hard function of Theorem 22. We will use the following (standard) combinatorial designs.

▶ **Claim 23** (NW Designs [19])**.** *For any positive integers n, α, there exists an efficiently computable family of sets $S_1, \ldots, S_n$ such that*

- $S_i \subset [r]$, $\forall i \in [n]$, *where* $r = n^{2/(\alpha+1)}$,
- $|S_i| = \ell = n^{1/(\alpha+1)}$, $\forall i \in [n]$, *and*
- $|S_i \cap S_j| \leq \alpha$, $\forall i, j \in [n]$ *such that* $i \neq j$.

Proof Theorem 21. For $\ell = n^{1/(\alpha+1)}$, let $f\colon \{0,1\}^\ell \to \{0,1\}$ be the hard function for degree-$(\alpha \cdot \Delta)$ PTF circuits from Theorem 22. By Theorem 22 and assuming E is a sufficiently large constant, we have that for any degree-$(\alpha \cdot \Delta)$ PTF circuit D on ℓ variables of size at most s,

$$\mathbf{Pr}_{z \sim \{0,1\}^\ell}[D(z) = f(z)] \leq \frac{1}{2} + \epsilon/n. \tag{7}$$

Let $S_1, \ldots, S_n$ be the sets from Claim 23. Define the generator $G_{\alpha,\Delta}\colon \{0,1\}^r \to \{0,1\}^n$ as follows:

$$G_{\alpha,\Delta}(y) = f(y|_{S_1}), \ldots, f(y|_{S_n}),$$

where, for $i \in [n]$, $y|_{S_i}$ denotes the substring of y indexed by the set S_i.

Toward a contradiction, suppose

$$|\mathbf{Pr}_{x \sim \{0,1\}^n}[C(x) = 1] - \mathbf{Pr}_{y \sim \{0,1\}^r}[C(G_{\alpha,\Delta}(y)) = 1]| > \epsilon. \tag{8}$$

By a standard argument via "reduction from distinguishing to predicting" as in [19], Equation (8) implies that there exist an $i \in [n]$, and bits $b_{i+1}, \ldots, b_n \in \{0,1\}$, such that

$$\mathbf{Pr}_{z \sim \{0,1\}^\ell}[C'(h_1(z), \ldots, h_i(z), b_{i+1}, \ldots, b_n) = f(z)] > 1/2 + \epsilon/n, \tag{9}$$

where

- $C' = C$ or $C' = \neg C$, and
- $h_1, \ldots, h_i$ are Boolean functions such that each depends on at most α bits of its input z.

First, note that each gate in C' is always a PTF of degree at most Δ. Next, observe that every Boolean function that depends on at most α variables can be computed by a multi-linear polynomial of degree at most α over the reals. Replacing our functions $h_1, \ldots, h_i$ with such degree α polynomials $p_1, \ldots, p_i$ inside C', we get

$$C'(p_1(z), \ldots, p_i(z), b_{i+1}, \ldots, b_n).$$

Now we can we merge the polynomials p_i's into every PTF gate in the circuit that reads from them. This yields a new circuit with *exactly the same* number of gates, and of degree at most $\alpha \cdot \Delta$. Denote this new circuit by C''. Note that C'' is a degree-$(\alpha \cdot \Delta)$ PTF circuit on ℓ variables of size at most s. By Equation (9), this PTF circuit C'' computes the function f with probability greater than $1/2 + \epsilon/n$, contradicting Equation (7). ◀

Then the PRG in Corollary 5 for PTFs can be obtained from the result in Theorem 21 by picking

$$\alpha = \frac{2 \log n}{L \cdot (\sqrt{\Delta \cdot \log n + 2 \log \log(1/\epsilon)})} - 1 \leq \frac{2}{L} \cdot \sqrt{\log n / \Delta},$$

where L is a sufficiently large constant. For this value of α, we get that the PRG in Theorem 21 fools any degree-Δ PTF circuit of size at least 1, and has seed length at most $\exp\left(O\left(\sqrt{\Delta \cdot \log n}\right)\right) \cdot \log^2(1/\epsilon)$.

High Order Random Walks: Beyond Spectral Gap

Tali Kaufman[1]
Department of Computer Science, Bar-Ilan University, Ramat Gan, Israel
kaufmant@mit.edu

Izhar Oppenheim
Department of Mathematics, Ben-Gurion University of the Negev,
P.O. Box 653, Be'er-Sheva, Israel
izharo@bgu.ac.il

Abstract

We study high order random walks on high dimensional expanders on simplicial complexes (i.e., hypergraphs). These walks walk from a k-face (i.e., a k-hyperedge) to a k-face if they are both contained in a $k+1$-face (i.e, a $k+1$ hyperedge). This naturally generalizes the random walks on graphs that walk from a vertex (0-face) to a vertex if they are both contained in an edge (1-face).

Recent works have studied the spectrum of high order walks operators and deduced fast mixing. However, the spectral gap of high order walks operators is inherently small, due to natural obstructions (called coboundaries) that do not happen for walks on expander graphs.

In this work we go beyond spectral gap, and relate the expansion of a function on k-faces (called k-cochain, for $k = 0$, this is a function on vertices) to its structure.

We show a Decomposition Theorem: For every k-cochain defined on high dimensional expander, there exists a decomposition of the cochain into i-cochains such that the square norm of the k-cochain is a sum of the square norms of the i-chains and such that the more weight the k-cochain has on higher levels of the decomposition the better is its expansion, or equivalently, the better is its shrinkage by the high order random walk operator.

The following corollaries are implied by the Decomposition Theorem:

- We characterize highly expanding k-cochains as those whose mass is concentrated on the highest levels of the decomposition that we construct. For example, a function on edges (i.e. a 1-cochain) which is locally thin (i.e. it contains few edges through every vertex) is highly expanding, while a function on edges that contains all edges through a single vertex is not highly expanding.
- We get *optimal* mixing for high order random walks on Ramanujan complexes. Ramanujan complexes are recently discovered bounded degree high dimensional expanders. The optimality in their mixing that we prove here, enable us to get from them **more efficient Two-Layer-Samplers** than those presented by the previous work of Dinur and Kaufman.

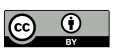
2012 ACM Subject Classification Theory of computation → Random walks and Markov chains, Mathematics of computing → Spectra of graphs, Mathematics of computing → Hypergraphs

Keywords and phrases High Dimensional Expanders, Simplicial Complexes, Random Walk

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.47

Related Version `https://arxiv.org/abs/1707.02799`

[1] This work was partially funded by ERC grant no. 336283 and BSF grant no. 2012256



1 Introduction

In this work we study high order random walks on bounded degree simplicial complexes (i.e., hypergraphs). These walks walk from a k-face (i.e., a k-hyperedge) to a k-face if they are both contained in a $(k+1)$-face (i.e, a $k+1$ hyperedge). This naturally generalizes the random walks on graphs that walk from a vertex (0-face) to a vertex if they are both contained in an edge (1-face). Roughly speaking, we are interested in walks that converge fast to a uniform distribution on the k-faces. For obtaining such fast convergence, a necessary condition is that the k-faces are connected. Namely, every pair of k-faces is connected by a path that is composed of $(k+1)$-faces (e.g., in graph every pair of vertices is connected by a path composed of edges). This high order connectivity is a necessary condition for the high order walk to converge. High order connectivity is hard to achieve in bounded degree complexes (or hypergraphs). For example, a random bounded degree simplicial complex (i.e., a complex chosen with uniform distribution from the set of all complexes with the same number of vertices and the same bound on the degree) does not have the high order connectivity property. In fact, there are only few currently known bounded degree complexes which do have this high order connectivity property. However, high order connectivity only ensures the convergence of high order random walk to a uniform distribution. For fast convergence, we need to require some form of high dimensional expansion. It turns out, that fast mixing of its high order random walk on a simplicial complex can be deduced its links structure. Links are local neighborhoods of faces in the complex (e.g. local neighborhoods of vertices, edges etc). A conclusion of this work, as well as previous works on high order random walks, is that requiring that ALL links (i.e. the local neighborhoods) are expanding implies fast mixing of high order random walks (we will discuss links and their expanding properties in detail momentarily). As said, random bounded degree complexes are not even connected, let alone their links are not good expanders.

In this paper we study high order random walks of simplicial complexes whose links are expanding (we call them local spectral expanders). High order random walks are strongly related to PCP agreement tests; direct product testing and direct sum testing [3]. This relation influenced, in part, the study of high order random walks.

The focus of previous works [8, 3] was to bound the second largest eigenvalue (in an absolute value) of the high order walk operator in complexes whose links are good spectral expanders. Namely, previous works have shown that in complexes with links that are good spectral expanders, every k-cochain that is orthogonal to the constant functions is shrinked by the k-order random walk operator M_k^+. I.e., by the walk that walk from a k-face to a k-face through $(k+1)$-face. The shrinkage rate is immediately determined by the second eigenvalue of the walk operator. However, there are natural obstructions (such as coboundaries) that prevents very large spectral gap of the walk operator.

It could well be the case that k-cochains with some specific structures are shrinked much better (or equivalently expand much better) than the bound obtained by spectral gap. This is similar in spirit to the small set expansion question in, say, the noisy hypercube [2], where the noisy hypercube is not a good expander so we can not say that general sets expand well; However, methods beyond spectral gap enabled showing that small sets of the noisy hypercube expand very well; this is similar to our goal here.

The focus of this work is to relate the *structure* of a k-cochain ϕ to its expansion, or equivalently, to the amount of its shrinkage by the random walk operator M_k^+, in complexes that are local spectral expanders. We provide a decomposition theorem that relates the amount of shrinkage of a k-cochain to the structure of the i-cochains to which it

decomposes $0 \leq i \leq k$. Specifically, we decompose ϕ into i-cochains, $0 \leq i \leq k$, such that $||\phi||^2 = \sum_{i=1}^{k} ||\phi_i||^2$ and show that the more weight ϕ has on the top levels the better is its shrinkage by the k-order random walk operator.

In particular, we derive the following conclusions from our decomposition theorem:

- We characterize k-cochains which do not expand a lot as those whose mass is concentrated on the lower levels of the decomposition that we construct. In particular, we show that "locally thin" binary cochains shrink dramatically by the high order random walk operator on a local spectral high dimensional expander. A binary k-cochain is "locally thin" if the degree of the cochain in each $(k-1)$-face is small. For example, a function on edges (i.e. a 1-cochain) which contains few edges through every vertex is locally thin, and hence highly expanding, while a function on edges that contains all edges through a single vertex is not highly expanding.
- We derive an *optimal* bound on the second eigenvalue (in an absolute value) of the high order random walk operator M_k^+ of complexes whose links are good *one sided spectral expanders*. Recent work of [3] have shown optimal bound on the second eigenvalue (in an absolute value) of M_k^+ for complexes whose links are good *two sided spectral expanders*. Thus, we get optimal bound on high order walks on Ramanujan complexes [9] which are one-sided local-spectral expanders, but NOT two sided spectral expanders. Ramanujan complexes are one of the few currently known examples of bounded degree complexes which are one-sided local-spectral expanders.

  Since the links of the well studies Ramanujan complexes are one-sided spectral expanders (but NOT two sided spectral expanders), the result of [3] does not apply to the Ramanujan complexes but only for complexes obtained from Ramanujan complexes (e.g. a k-skeleton of k^2 Ramanujan complex). Our result is the first result obtaining *optimal* bounds of the spectrum of high order random walks on the Ramanujan complexes themselves. The optimality in the mixing of the Ramanujan complexes that we prove here, enable us to get from them **more efficient Two-Layer-Samplers** than those presented by [3]. In the following we discuss the notion of two layers sampler (that was introduced by [3]), and discuss the more efficient two-layers-samplers that we get in this work.

1.1 More efficient Two-Layer-Samplers implied by this work

The work of [3] that studied agreement expansion and agreement tests introduced to following generalization of a sampler that is called a *two-layer-sampler*

▶ **Definition 1.** An infinite family of tripartite incidence graphs $\{G(L \cup R \cup W, E_1 \cup E_2)\}_l$ is called a family of *two-layer-samplers* if there are constants $1 \leq k \leq n$, $\gamma > 0$ such that the following conditions hold:

- $L = [l]$, $R \subseteq \binom{[l]}{k}$, $W \subseteq \binom{[l]}{n}$ such that there is an edge between $r \in R$ and $x \in L$ if $x \subset r$ and such that there is an edge between $w \in W$ and $r \in R$ if $r \subset w$.
- $|R| + |W| = O(l)$ where the constants depend only on k, n, γ.
- G has the following double expansion property:

$$(\lambda(G(L,R)))^2 \leq \frac{1}{k} + \gamma \text{ and } (\lambda(G(R,W)))^2 \leq \frac{k}{n} + \gamma,$$

where $G(L,R)$, $G(R,W)$ are the respective bipartite graphs and λ is the second largest normalized singular value of the appropriate transition matrix.

The work of [3] has shown that a family of two-layer-samplers as above with sets of size 1, k, n could be derived from the 0,$k-1$,$n-1$ faces of a simplicial complexes in which the $n-1$-order random walk has *optimal* mixing. For obtaining complexes with such optimal mixing of $(n-1)$-order walks [3] used the Ramanujan complexes of dimension $(n-1)^2$. Thus, the 1, k, n-sets that appear in the two-layer samplers that they construct correspond to 0,$k-1$,$n-1$ faces in a Ramanujan complexes of dimension $(n-1)^2$. Our result here implies optimal mixing of $(n-1)$-random walks in Ramanujan complexes of dimension $n-1$ themselves (i.e., we do not need to take Ramanujan complexes of dimension $(n-1)^2$ in order to argue about optimality of $d-1$-random walks.). Thus, we get a two-layer sampler whose 1, k, n sets correspond to 0,$k-1$,$n-1$ faces in a Ramanujan complex of dimension $(n-1)$ (and not dim $(n-1)^2$). The fact that we use Ramanujan complexes of smaller dimension implies that our sampler uses sets L, R, W which are smaller than those obtained by [3] and hence our two-layer-samplers are more efficient.

1.2 On simplicial complexes and localization

A pure n-dimensional simplicial complex X is a simplicial complex in which every simplex is contained in an n-dimensional simplex. In other words, it is an $(n+1)$-hypergraph with a closure property: for every hyperedge in the hypergraph, all of its subsets are also hyperedges in the hypergraph. The sets with $i+1$ elements are denoted $X(i)$, $0 \leq i \leq n$. The *one-skeleton* of the complex X is its underlying graph obtained by $X(0) \cup X(1)$. A set $\tau \in X(i)$ is called a *face*. The *link* of τ denoted X_τ is the complex obtained by taking all faces in X that contain τ and removing τ from them. Thus, if τ is of dimension i (i.e. $\tau \in X(i)$) then X_τ is of dimension $n-i-1$.

For every $-1 \leq i \leq n-2$, the one skeleton of X_τ is a graph. Its second largest eigenvalue is μ_τ; its smallest eigenvalue is ν_τ.

▶ **Definition 2** (One sided local spectral expander)**.** A pure n-dimensional complex X is a *one-sided λ-local-spectral expander* if for every $-1 \leq i \leq n-2$, and for every $\tau \in X(i)$, $\mu_\tau \leq \lambda$.

▶ **Definition 3** (Two sided local spectral expander)**.** A pure n-dimensional complex X is a *two-sided λ-local-spectral expander* if for every $-1 \leq i \leq n-2$, and for every $\tau \in X(i)$, $-\lambda \leq \nu_\tau$ and $\mu_\tau \leq \lambda$.

1.3 A decomposition theorem for high order random walks and its implications

We study the random walk operators: M_k^+, corresponding to the walk from a k-face to a k-face through $k+1$ face. (For exact definition see Section 2). We normalize the operator so that the largest eigenvalue is 1.

In this paper we show the following decomposition theorem that for one-sided λ-local-spectral expanders:

▶ **Theorem 4** (decomposition Theorem, informal, for formal see Theorem 18)**.** *Given a pure n-dimensional one-sided λ-local-spectral expander X. For a k-cochain ϕ ($k \leq n-1$) orthogonal to the constant functions there exist i-cochains ϕ_i for every $0 \leq i \leq k$ such that $||\phi||^2 = \sum_{i=0}^{k} ||\phi_i||^2$ such that*

$$\langle M_k^+ \phi, \phi \rangle \leq \sum_{i=0}^{k} \frac{k+1-i}{k+2} ||\phi_i||^2 + (k+1)\lambda ||\phi||^2.$$

As a corollary of the decomposition theorem we derive optimal bounds on the second largest eigenvalue (in an absolute value) of M_k^+ for X which is one-sided λ-local-spectral expander. This result is stronger than [3] that applies only for two-sided λ-local-spectral expander.

▶ **Theorem 5** (Bounding the second eigenvalue of the k-walk Theorem, informal, for formal see Theorem 20)**.** *Given a pure n-dimensional one-sided λ-local-spectral expander X. For a k-cochain ϕ ($k \leq n-1$) orthogonal to the constant functions:*

$$\|M_k^+\phi\| \leq \left((1 - \frac{1}{k+2}) + (k+1)\lambda\right) ||\phi||.$$

This result improves on previous results: in [8] a similar result is given, but the bound on $\|M_k^+\phi\|$ is less tight (the bound is $((1 - \frac{1}{(k+2)^2}) + O((k+1)\lambda)||\phi||)$. In [3], a similar bound is given, but under the stronger assumption that X is a two-sided λ-local-spectral expander. The seemingly mild improvement that [3] achieves over [8] is crucial for their application. However, as the Ramanujan complexes are only one-sided λ-local-spectral expanders, the result of [3] does not apply to the Ramanujan complexes themselves but only to other complexes that could be built based on them.

All of the results discussed before are proven with respect to the lazy random walk operator M_k^+. The operator is called lazy since when the walk is located on a certain k-face is has non-zero probability that the next step will stay on the same k-face and will not move. Similarly one can define a a non-lazy random walk operator (see Definition 8). We show that for complexes that are two-sided λ-local-spectral expanders, similar results that we obtained for the lazy ransom walks could be obtained also for the non-lazy random walk.

1.4 On small set expansion phenomenon, the Grassmann complex and our work

As we have explained above, we study the amount of shrinkage of a k-cochain by the random walk operator M_k^+. Our motivation is to go beyond spectral gap and to related the shrinkage of a k-cochain by the operator, to its structure. Similar questions are asked in the study of small set expansion in the noisy hypercube [2]. Recently it was shown that studying the structure of non expanding k-cochains of the Grassmann complex is strongly related to the "2-to-1 games Conjecture" [4, 5], which is a weaker form of the famous Unique Game Conjecture. Our work here, is of the same flavor. However, instead of working with a specific complex (e.g the Grassmann) we work with simplicial complexes, whose links are good spectral expanders. We characterise non expanding k-cochains as those whose mass is concentrated on the lower levels of the decomposition that we construct.

2 Definitions and notations

Let X be a pure n-dimensional finite simplicial complex. For $-1 \leq k \leq n$, we denote $X(k)$ to be the set of all k-simplices in X ($X(-1) = \{\emptyset\}$). A weight function m on X is a function: $m : \bigcup_{-1 \leq k \leq n} X(k) \to \mathbb{R}^+$, such that for every $-1 \leq k \leq n-1$ and for every $\tau \in X(k)$ we have that $m(\tau) = \sum_{\sigma \in X(k+1)} m(\sigma)$. By this definition, it is clear the m is determined by the values in takes in $X(n)$. A simplicial complex with a weight function will be called a weighted simplcial complex. For a pure n-dimensional simplicial complex there is a natural weight function m_h which we call the homogeneous weight function (since it give the value 1

to each n-dimensional simplex) defined as follows:

$$\forall \tau \in X(k), m_h(\tau) = (n-k)!|\{\eta \in X(n) : \tau \subseteq \eta\}|.$$

We leave it for the reader to verify that m_h is indeed a weight function.

Throughout this article, X is a pure n-dimensional finite weighted simplicial complex with a weight function m.

For $-1 \leq k \leq n-1$, we denote $C^k(X, \mathbb{R})$ to be the set of all functions $\phi : X(k) \to \mathbb{R}$. Abusing the terminology, we will call the space $C^k(X, \mathbb{R})$ the space of non-oriented cochains. On $C^k(X, \mathbb{R})$ define the following inner-product:

$$\forall \phi, \psi \in C^k(X, \mathbb{R}), \langle \phi, \psi \rangle = \sum_{\sigma \in X(k)} m(\sigma)\phi(\sigma)\psi(\sigma).$$

Denote by $\|.\|$ the norm induced by this inner-product.

3 Upper and lower random walks

Let X be a pure n-dimensional finite weighted simplicial complex with a weight function m. We will define the following random walks on simplices of X:

▶ **Definition 6.** For $0 \leq k \leq n-1$, the upper random walk on k-simplices is defined by the transition probability matrix $M_k^+ : X(k) \times X(k) \to \mathbb{R}$:

$$M_k^+(\tau, \tau') = \begin{cases} \frac{1}{k+2} & \tau = \tau' \\ \frac{m(\tau \cup \tau')}{(k+2)m(\tau)} & \tau \cup \tau' \in X(k+1) \\ 0 & \text{otherwise} \end{cases}.$$

▶ **Definition 7.** For $0 \leq k \leq n$, the lower random walk on k-simplices is defined by the transition probability matrix $M_k^- : X(k) \times X(k) \to \mathbb{R}$:

$$M_k^-(\tau, \tau') = \begin{cases} \sum_{\eta \in X(k-1)} \frac{m(\tau)}{(k+1)m(\eta)} & \tau = \tau' \\ \frac{m(\tau')}{(k+1)m(\tau \cap \tau')} & \tau \cap \tau' \in X(k-1) \\ 0 & \text{otherwise} \end{cases}.$$

We leave it to the reader to check that those are in fact transition probability matrix, i.e., that for every τ, $\sum_{\tau'} M_k^{\pm}(\tau, \tau') = 1$. We note that both the random walks defined above are lazy in the sense that $M^{\pm}(\tau, \tau) \neq 0$. In the case of the upper random walk, one can easily define a non lazy random walk as follows:

▶ **Definition 8.** For $0 \leq k \leq n-1$, the non-lazy upper random walk on k-simplices is defined by the transition probability matrix $(M')_k^+ : X(k) \times X(k) \to \mathbb{R}$:

$$(M')_k^+ = \frac{k+2}{k+1}\left(M_k^+ - \frac{1}{k+2}I\right) = \frac{k+2}{k+1}M_k^+ - \frac{1}{k+1}I.$$

It is standard to view $M_k^{\pm}, (M')_k^+$ as averaging operators on $C^k(X, \mathbb{R})$ and we will not make the distinction between the transition probability matrix and the averaging operator it induces.

It is worth noting that M_0^- and $(M')_0^+$ are familiar operators/matrices: M_0^- is a projection on the space of the constant functions (on vertices) with respect to the inner-product defined above, and $(M')_0^+$ is the weighted (normalized) adjacency matrix of the 1-skeleton of X.

4 The signless differential

▶ **Definition 9.** For $-1 \leq k \leq n-1$, the signless k-differential is an operator $d_k : C^k(X,\mathbb{R}) \to C^{k+1}(X,\mathbb{R})$ defined as:

$$\forall \phi \in C^k(X,\mathbb{R}), \forall \sigma \in X(k+1), d_k\phi(\sigma) = \sum_{\tau \subset \sigma, \tau \in X(k)} \phi(\tau).$$

Define $(d_k)^* : C^{k+1}(X,\mathbb{R}) \to C^k(X,\mathbb{R})$ to be the adjoint operator to d_k, i.e., the operator such that for every $\phi \in C^k(X,\mathbb{R}), \psi \in C^{k+1}(X,\mathbb{R})$, $\langle d_k\phi, \psi\rangle = \langle \phi, (d_k)^*\psi\rangle$.

▶ Remark. We note that the signless differential is not a differential in the usual sense, since $d_{k+1}d_k \neq 0$. The name signless differential stems from the fact that this is the operator we will use in lieu of the differential in our setting (note that since our non-oriented cochains are defined without using orientation of simplices, we cannot use the usual differential).

Below, we will usually omit the index of signless differential and its adjoint and just denote d, d^* where k will be implicit.

▶ **Lemma 10.** *For $-1 \leq k \leq n-1$, $d^* : C^{k+1}(X,\mathbb{R}) \to C^k(X,\mathbb{R})$ is the operator*

$$\forall \psi \in C^{k+1}(X,\mathbb{R}), \forall \tau \in X(k), d^*\psi(\tau) = \sum_{\sigma \in X(k+1), \tau \subset \sigma} \frac{m(\sigma)}{m(\tau)}\psi(\sigma).$$

Proof. Let $\phi \in C^k(X,\mathbb{R})$ and $\psi \in C^{k+1}(X,\mathbb{R})$. Then

$$\langle d\phi, \psi\rangle = \sum_{\sigma \in X(k+1)} m(\sigma) d\phi(\sigma)\psi(\sigma) = \sum_{\sigma \in X(k+1)} m(\sigma) \sum_{\tau \in X(k), \tau \subset \sigma} \phi(\tau)\psi(\sigma) =$$
$$\sum_{\tau \in X(k)} \phi(\tau) \sum_{\sigma \in X(k+1), \tau \subset \sigma} m(\sigma)\psi(\sigma) =$$
$$\sum_{\tau \in X(k)} m(\tau)\phi(\tau) \left(\sum_{\sigma \in X(k+1), \tau \subset \sigma} \frac{m(\sigma)}{m(\tau)}\psi(\sigma) \right) = \langle \phi, d^*\psi\rangle.$$

◀

▶ **Corollary 11.** *For $0 \leq k \leq n-1$ and $\phi \in C^k(X,\mathbb{R})$, $d^*d\phi = (k+2)M^+\phi$ and $dd^*\phi = (k+1)M^-\phi$.*

Proof. Let $\phi \in C^k(X,\mathbb{R})$ and $\tau \in X(k)$, then

$$d^*d\phi(\tau) = \sum_{\sigma \in X(k+1), \tau \subset \sigma} \frac{m(\sigma)}{m(\tau)} d\phi(\sigma) = \sum_{\sigma \in X(k+1), \tau \subset \sigma} \frac{m(\sigma)}{m(\tau)} \sum_{\tau' \in X(k), \tau' \subset \sigma} \phi(\tau') =$$
$$\sum_{\sigma \in X(k+1), \tau \subset \sigma} \frac{m(\sigma)}{m(\tau)} \sum_{\tau' \in X(k), \tau' \subset \sigma, \tau' \neq \tau} \phi(\tau') + \sum_{\sigma \in X(k+1), \tau \subset \sigma} \frac{m(\sigma)}{m(\tau)} \phi(\tau).$$

Note that

$$\sum_{\sigma \in X(k+1), \tau \subset \sigma} \frac{m(\sigma)}{m(\tau)} \phi(\tau) = \frac{m(\tau)}{m(\tau)} \phi(\tau) = \phi(\tau).$$

Also note that

$$\sum_{\sigma\in X(k+1),\tau\subset\sigma}\frac{m(\sigma)}{m(\tau)}\sum_{\tau'\in X(k),\tau'\subset\sigma,\tau'\neq\tau}\phi(\tau')=$$
$$\sum_{\sigma\in X(k+1),\tau\subset\sigma}\sum_{\tau'\in X(k),\tau'\subset\sigma,\tau'\neq\tau}\frac{m(\tau'\cup\tau)}{m(\tau)}\phi(\tau')=\sum_{\tau'\in X(k),\tau\cup\tau'\in X(k+1)}\frac{m(\tau\cup\tau')}{m(\tau)}\phi(\tau').$$

Therefore

$$d^*d\phi(\tau)=\phi(\tau)+\sum_{\tau'\in X(k),\tau\cup\tau'\in X(k+1)}\frac{m(\tau\cup\tau')}{m(\tau)}\phi(\tau')=(k+2)M^+\phi(\tau).$$

Similarly,

$$dd^*\phi(\tau)=\sum_{\eta\in X(k-1),\eta\subset\tau}d^*\phi(\eta)=\sum_{\eta\in X(k-1),\eta\subset\tau}\sum_{\tau'\in X(k),\eta\subset\tau'}\frac{m(\tau')}{m(\eta)}\phi(\tau')=$$
$$\sum_{\eta\in X(k-1),\eta\subset\tau}\sum_{\tau'\in X(k),\tau'\neq\tau,\eta\subset\tau'}\frac{m(\tau')}{m(\eta)}\phi(\tau')+\sum_{\eta\in X(k-1),\eta\subset\tau}\frac{m(\tau)}{m(\eta)}\phi(\tau)=$$
$$\sum_{\eta\in X(k-1),\eta\subset\tau}\sum_{\tau'\in X(k),\tau'\cap\tau=\eta}\frac{m(\tau')}{m(\tau\cap\tau')}\phi(\tau')+\sum_{\eta\in X(k-1),\eta\subset\tau}\frac{m(\tau)}{m(\eta)}\phi(\tau)=$$
$$\sum_{\tau'\in X(k),\tau\cap\tau'\in X(k-1)}\frac{m(\tau')}{m(\tau\cap\tau')}\phi(\tau')+\sum_{\eta\in X(k-1),\eta\subset\tau}\frac{m(\tau)}{m(\eta)}\phi(\tau)=(k+1)M^-\phi(\tau).$$

◀

5 Links and localization

Let X be a pure n-dimensional finite simplicial complex with a weight function m. Recall that for $-1\leq k\leq n-1$, $\tau\in X(k)$, the link of τ, denoted X_τ, is a pure $(n-k-1)$-simplicial complex defined as:

$$\eta\in X_\tau(l)\Leftrightarrow\eta\in X(l)\text{ and }\tau\cup\eta\in X(k+l+1).$$

On X_τ we define the weight function m_τ induced by m as

$$m_\tau(\eta)=m(\tau\cup\eta).$$

Using this weight function the inner-product and the norm on $C^l(X_\tau,\mathbb{R})$ are defined as above. The operators $M^\pm_{\tau,l}$, $(M')^+_{\tau,l}$ and d_τ, d^*_τ are also defined on $C^l(X_\tau,\mathbb{R})$ as above.

Given a cochain $\phi\in C^l(X,\mathbb{R})$ and a simplex $\tau\in X(k)$ with $-1\leq k<l$, we define the localization of ϕ on X_τ, denoted ϕ_τ as a cochain $\phi_\tau\in C^{l-k-1}(X_\tau,\mathbb{R})$ defined as

$$\phi_\tau(\eta)=\phi(\tau\cup\eta).$$

The key observation (which was initially due to Garland [6], but is now considered standard – see [1], [7]) is that the norm of ϕ, $d^*\phi$, and $d\phi$ can be calculated via their localizations. Also, the work of the second named author implies that the local spectral information needed to bound $d\phi$ can be deduced from the spectral information of the 1-dimensional links. The proof are somewhat technical and therefore below we will just state the results and give the proofs in the appendix.

▶ **Proposition 12.** *Let $-1 \leq k < l \leq n$ and let $\phi \in C^l(X, \mathbb{R})$, then*

$$\binom{l+1}{k+1}\|\phi\|^2 = \sum_{\tau \in X(k)} \|\phi_\tau\|^2 \text{ and } \binom{l}{k+1}\|d^*\phi\|^2 = \sum_{\tau \in X(k)} \|d^*_\tau \phi_\tau\|^2.$$

Also, if $l < n$, then

$$\|d\phi\|^2 = \sum_{\tau \in X(l-1)} \left(\|d_\tau \phi_\tau\|^2 - \frac{l}{l+1}\|\phi_\tau\|^2 \right).$$

As a result of Proposition 12 we deduce the following:

▶ **Proposition 13.** *Let $-1 \leq k \leq n-1$ and let $\phi \in C^k(X, \mathbb{R})$, then*

$$\|d\phi\|^2 = \|d^*\phi\|^2 + \|\phi\|^2 + \sum_{\tau \in X(k-1)} \langle (M')^+_{\tau,0}(I - M^-_{\tau,0})\phi_\tau, \phi_\tau \rangle.$$

In light of the above Proposition, we will want to bound the expression

$$\sum_{\tau \in X(k-1)} \langle (M')^+_{\tau,0}(I - M^-_{\tau,0})\phi_\tau, \phi_\tau \rangle,$$

using spectral information about X. To make this precise, we will recall/define the following. For $0 \leq k \leq n-1$ and $\tau \in X(k-1)$, recall that by Corollary 11, $(M')^+_{\tau,0} = d^*_\tau d_\tau - I$ and therefore the eigenvalues of $(M')^+_{\tau,0}$ are real. Denote μ_τ to be the second largest eigenvalue of $(M')^+_{\tau,0}$ and ν_τ to be the smallest eigenvalue of $(M')^+_{\tau,0}$. Note that if 1-skeleton of X_τ is connected, then for every eigenfunction $\psi \in C^0(X_\tau, \mathbb{R})$, if $\psi \perp \operatorname{Im} M^-_{\tau,0}$, then $(M')^+_{\tau,0}\psi = \mu\psi$ with $\nu_\tau \leq \mu \leq \mu_\tau < 1$. Denote

$$\mu_k = \max_{\tau \in X(k-1)} \mu_\tau, \nu_k = \min_{\tau \in X(k-1)} \nu_\tau.$$

▶ **Lemma 14.** *For every $0 \leq k \leq n-1$ and every $\phi \in C^k(X, \mathbb{R})$ we have that*

$$\sum_{\tau \in X(k-1)} \langle (M')^+_{\tau,0}(I - M^-_{\tau,0})\phi_\tau, \phi_\tau \rangle \leq (k+1)\mu_k \|\phi\|^2,$$

and

$$\sum_{\tau \in X(k-1)} \langle (M')^+_{\tau,0}(I - M^-_{\tau,0})\phi_\tau, \phi_\tau \rangle \geq (k+1)\nu_k \|\phi\|^2$$

We recall the following definition from the introduction:

▶ **Definition 15** (Local spectral expander)**.** A n-dimensional complex X is a *one-sided λ-local-spectral expander* if for every $0 \leq k \leq n-1$, $\mu_k \leq \lambda$. A n-dimensional complex X is a *two-sided λ-local-spectral expander* if for every $0 \leq k \leq n-1$, $-\lambda \leq \nu_k$ and $\mu_k \leq \lambda$.

Therefore for one-sided λ-local-spectral expanders, λ can be used to bound $\sum_{\tau \in X(k-1)} \langle (M')^+_{\tau,0}(I - M^-_{\tau,0})\phi_\tau, \phi_\tau \rangle$ from above (and from below in the case of two-sided λ-local-spectral expanders).

We recall the following result appearing in [11][Lemma 5.1] (see also [10][Proposition 3.7]):

▶ **Lemma 16.** *Let X be a weighted pure n-dimensional simplicial complex, such that all the links of X of dimension ≥ 1 (including X itself) are connected, then for every $0 \leq k \leq n-2$,*

$$\mu_k \leq \frac{\mu_{k+1}}{1 - \mu_{k+1}} \text{ and } \nu_k \geq \frac{\nu_{k+1}}{1 - \nu_{k+1}}.$$

6 Decomposition theorem for upper random walks

For every $0 \leq k \leq n-1$, we denote $C_0^k(X,\mathbb{R})$ to be

$$C_0^k(X,\mathbb{R}) = \{\phi \in C^k(X,\mathbb{R}) : \sum_{\sigma \in X(k)} m(\sigma)\phi(\sigma) = 0\}.$$

Let $\mathbb{1}_k$ to be the constant 1 function in $C^k(X,\mathbb{R})$, then by definition for every $\phi \in C_0^k(X,\mathbb{R})$, we have that $\langle \phi, \mathbb{1}_k \rangle = \sum_{\sigma \in X(k)} m(\sigma)\phi(\sigma) = 0$, and one can see that $C^k(X,\mathbb{R})$ has the orthogonal decomposition $C^k(X,\mathbb{R}) = \text{span}\{\mathbb{1}_k\} \oplus C_0^k(X,\mathbb{R})$. It is easy to see that $M_k^{\pm}\mathbb{1}_k = \mathbb{1}_k$ and since, by Corollary 11, M_k^+, M_k^- are self-adjoint operators, is follows that $M_k^{\pm}(C_0^k(X,\mathbb{R})) \subseteq C_0^k(X,\mathbb{R})$.

▶ **Lemma 17.** *For $0 \leq k \leq n-1$, $\ker((d_{k-1})^*) \subseteq C_0^k(X,\mathbb{R})$ and*

$$\forall \psi \in C^{k-1}(X,\mathbb{R}), d_{k-1}\psi \in C_0^k(X,\mathbb{R}) \Rightarrow \psi \in C_0^{k-1}(X,\mathbb{R}).$$

Proof. We note that by definition $d_{k-1}\mathbb{1}_{k-1} = (k+1)\mathbb{1}_k$, and, by Lemma 10, $(d_{k-1})^*\mathbb{1}_k = \mathbb{1}_{k-1}$. Therefore for every $\phi \in \ker((d_{k-1})^*)$, we have that

$$0 = \langle (d_{k-1})^*\phi, \mathbb{1}_{k-1}\rangle = \langle \phi, (d_{k-1})\mathbb{1}_{k-1}\rangle = (k+1)\langle \phi, (d_{k-1})\mathbb{1}_k\rangle \Rightarrow \phi \in C_0^k(X,\mathbb{R}).$$

Also, for every $\psi \in C^{k-1}(X,\mathbb{R})$ such that $d_{k-1}\psi \in C_0^k(X,\mathbb{R})$, we have that

$$0 = \langle d_{k-1}\psi, \mathbb{1}_k\rangle = \langle \psi, (d_{k-1})^*\mathbb{1}_k\rangle = \langle \psi, \mathbb{1}_{k-1}\rangle \Rightarrow \psi \in C_0^{k-1}(X,\mathbb{R}). \quad \blacktriangleleft$$

▶ **Theorem 18** (Decomposition Theorem)**.** *For every $0 \leq k \leq n-1$ and every $\phi \in C_0^k(X,\mathbb{R})$, there are $\phi^k \in C_0^k(X,\mathbb{R}), \phi^{k-1}, (\phi^{k-1})' \in C_0^{k-1}(X,\mathbb{R}), ..., \phi^0, (\phi^0)' \in C_0^0(X,\mathbb{R})$ such that if we denote $(\phi^k)' = \phi$, then the following holds:*

1. *For every $0 \leq i \leq k$, $\|(\phi^i)'\|^2 = \|\phi^i\|^2 + \|\phi^{i-1}\|^2 + ... + \|\phi^0\|^2$.*
2. *$\|d\phi\|^2 = \sum_{i=0}^k (k+1-i)\|\phi^i\|^2 + \sum_{i=0}^k \sum_{\tau \in X(i-1)} \langle (M')_{\tau,0}^+ (I - M_{\tau,0}^-)(\phi^i)'_\tau, (\phi^i)'_\tau\rangle$.*

Proof. We will prove the theorem by induction on k. For $k=0$ and $\phi \in C_0^0(X,\mathbb{R})$, we take $\phi^0 = \phi$ and check that the theorem holds for this choice.

1. This condition holds trivially.
2. We note that $\phi \in C_0^0(X,\mathbb{R})$ implies that $d^*\phi = 0$ and therefore this condition follows from Proposition 13.

Assume next that $k > 0$ and that the theorem holds for $k-1$. For $\phi \in C_0^k(X,\mathbb{R})$, we first decompose ϕ as $\phi = \phi^k + \phi'$, where $\phi^k \in \ker((d_{k-1})^*)$ and $\phi' \in (\ker(d_{k-1})^*))^\perp$. This is an orthogonal decomposition and therefore $\|\phi\|^2 = \|\phi^k\|^2 + \|\phi'\|^2$. Also, by Proposition 13,

$$\|d_k\phi\|^2 = \|\phi\|^2 + \|(d_{k-1})^*\phi'\|^2 + \sum_{\tau \in X(k-1)} \langle (M')_{\tau,0}^+ (I - M_{\tau,0}^-)\phi_\tau, \phi_\tau\rangle. \tag{1}$$

We note that $(\ker(d_{k-1})^*))^\perp = \text{Im}(d_{k-1})$ and therefore, by using Lemma 17, there is $\psi \in C_0^{k-1}(X,\mathbb{R})$ such that $d_{k-1}\psi = \phi'$. This yields that there is $\psi \in C_0^{k-1}(X,\mathbb{R})$, such that $\|d_{k-1}\psi\|^2 = \|\phi'\|^2$ and

$$\|d\phi\|^2 = \|\phi\|^2 + \|(d_{k-1})^*d_{k-1}\psi\|^2 + \sum_{\tau \in X(i-1)} \langle (M')_{\tau,0}^+ (I - M_{\tau,0}^-)\phi_\tau, \phi_\tau\rangle.$$

We recall that since $(d_{k-1})^*d_{k-1}$ is a self-adjoint operator, with non negative eigenvalues, $\sqrt{(d_{k-1})^*d_{k-1}}$ is the self-adjoint operator, with non negative eigenvalues defined as follows:

for every eigenfunction φ of $(d_{k-1})^* d_{k-1}$ with an eigenvalue μ, φ is an eigenfunction of $\sqrt{(d_{k-1})^* d_{k-1}}$ with the eigenvalue $\sqrt{\mu}$.

We will take $(\phi^{k-1})' = \sqrt{(d_{k-1})^* d_{k-1}}\psi$ and check that the theorem holds for this choice.

First, we note that, using corollary 11, $(d_{k-1})^* d_{k-1}(C_0^{k-1}(X,\mathbb{R})) \subseteq C_0^{k-1}(X,\mathbb{R})$, and therefore $\sqrt{(d_{k-1})^* d_{k-1}}(C_0^{k-1}(X,\mathbb{R})) \subseteq C_0^{k-1}(X,\mathbb{R})$, which implies that $(\phi^{k-1})' = \sqrt{(d_{k-1})^* d_{k-1}}\psi \in C_0^{k-1}(X,\mathbb{R})$.

Second, we note that

$$\|d_{k-1}\psi\|^2 = \langle (d_{k-1})^* d_{k-1}\psi, \psi\rangle = \langle \sqrt{(d_{k-1})^* d_{k-1}}\psi, \sqrt{(d_{k-1})^* d_{k-1}}\psi\rangle = \|\sqrt{(d_{k-1})^* d_{k-1}}\psi\|^2.$$

Therefore, $\sqrt{(d_{k-1})^* d_{k-1}}\psi \in C_0^{k-1}(X,\mathbb{R})$ and $\|\sqrt{(d_{k-1})^* d_{k-1}}\psi\| = \|\phi'\|$. This yields that

$$\|\phi\|^2 = \|\phi^k\|^2 + \|(\phi^{k-1})'\|^2,$$

and by the induction assumption

$$\|\phi\|^2 = \|\phi^k\|^2 + \|\phi^{k-1}\|^2 + ... + \|\phi^0\|^2. \tag{2}$$

Last, we note that

$$\|(d_{k-1})^*\phi'\|^2 = \|(d_{k-1})^* d_{k-1}\psi\|^2 = \langle (d_{k-1})^* d_{k-1}\psi, (d_{k-1})^* d_{k-1}\psi\rangle = \langle (d_{k-1})^* d_{k-1}\sqrt{(d_{k-1})^* d_{k-1}}\psi, \sqrt{(d_{k-1})^* d_{k-1}}\psi\rangle = \|d_{k-1}\sqrt{(d_{k-1})^* d_{k-1}}\psi\|^2 = \|d_{k-1}(\phi^{k-1})'\|^2.$$

Combining this with (1), we get that

$$\|d\phi\|^2 = \|\phi\|^2 + \|d_{k-1}(\phi^{k-1})'\|^2 + \sum_{\tau\in X(k-1)} \langle (M')^+_{\tau,0}(I - M^-_{\tau,0})\phi_\tau, \phi_\tau\rangle.$$

By the induction assumption,

$$\|d_{k-1}(\phi^{k-1})'\|^2 = \sum_{i=0}^{k-1}(k-i)\|\phi^i\|^2 + \sum_{i=0}^{k-1}\sum_{\tau\in X(i-1)} \langle (M')^+_{\tau,0}(I - M^-_{\tau,0})(\phi^i)'_\tau, (\phi^i)'_\tau\rangle.$$

Therefore

$$\|d\phi\|^2 = \|\phi\|^2 + \|d_{k-1}(\phi^{k-1})'\|^2 + \sum_{\tau\in X(k-1)} \langle (M')^+_{\tau,0}(I - M^-_{\tau,0})\phi_\tau, \phi_\tau\rangle =$$

$$\|\phi\|^2 + \sum_{i=0}^{k-1}(k-i)\|\phi^i\|^2 + \sum_{i=0}^{k-1}\sum_{\tau\in X(i-1)} \langle (M')^+_{\tau,0}(I - M^-_{\tau,0})(\phi^i)'_\tau, (\phi^i)'_\tau\rangle + \sum_{\tau\in X(k-1)} \langle (M')^+_{\tau,0}(I - M^-_{\tau,0})\phi_\tau, \phi_\tau\rangle =$$

$$\|\phi\|^2 + \sum_{i=0}^{k-1}(k-i)\|\phi^i\|^2 + \sum_{i=0}^{k}\sum_{\tau\in X(i-1)} \langle (M')^+_{\tau,0}(I - M^-_{\tau,0})(\phi^i)'_\tau, (\phi^i)'_\tau\rangle =$$

$$\sum_{i=0}^{k}(k+1-i)\|\phi^i\|^2 + \sum_{i=0}^{k}\sum_{\tau\in X(i-1)} \langle (M')^+_{\tau,0}(I - M^-_{\tau,0})(\phi^i)'_\tau, (\phi^i)'_\tau\rangle,$$

where the last equality is due to (2). ◀

▶ **Corollary 19.** *Let X be a pure n-dimensional weighted simplicial complex such that all the links of X of dimension ≥ 1 are connected (including X itself) and let $0 \leq k \leq n-1$. Then for every $\phi \in C_0^k(X,\mathbb{R})$, there are $\phi^k \in C_0^k(X,\mathbb{R}), \phi^{k-1} \in C_0^{k-1}(X,\mathbb{R}), ..., \phi^0 \in C_0^0(X,\mathbb{R})$, such that*

$$\|\phi\|^2 = \|\phi^k\|^2 + ... + \|\phi^0\|^2,$$

and

$$\|d\phi\|^2 \leq \sum_{i=0}^{k}(k+1-i+\sum_{j=i}^{k}(j+1)\mu_j)\|\phi^i\|^2,$$

$$\|d\phi\|^2 \geq \sum_{i=0}^{k}(k+1-i+\sum_{j=i}^{k}(j+1)\nu_j)\|\phi^i\|^2.$$

Proof. Let $\phi \in C_0^k(X,\mathbb{R})$ and $\phi^k \in C_0^k(X,\mathbb{R}), \phi^{k-1}, (\phi^{k-1})' \in C_0^{k-1}(X,\mathbb{R}), ..., \phi^0, (\phi^0)' \in C_0^0(X,\mathbb{R})$ as in the Decomposition Theorem. Then $\|\phi\|^2 = \|\phi^k\|^2 + ... + \|\phi^0\|^2$, and we will prove that $\|d\phi\|^2 \leq \sum_{i=0}^{k}(k+1-i+\sum_{j=i}^{k}(j+1)\mu_j)\|\phi^i\|^2$, (the proof of the second inequality is similar and therefore it is left to the reader).

Note that for every $0 \leq i \leq k$, we have by Lemma 14 that

$$\sum_{\tau \in X(i-1)} \langle (M')^+_{\tau,0}(I - M^-_{\tau,0})(\phi^i)'_\tau, (\phi^i)'_\tau \rangle \leq (i+1)\mu_i \|(\phi^i)'\|^2.$$

Therefore

$$\sum_{i=0}^{k}\sum_{\tau \in X(i-1)} \langle (M')^+_{\tau,0}(I - M^-_{\tau,0})(\phi^i)'_\tau, (\phi^i)'_\tau \rangle \leq \sum_{i=0}^{k}(i+1)\mu_i \sum_{j=0}^{i}\|\phi^j\|^2 = \sum_{j=0}^{k}\|\phi^j\|^2 \sum_{i=j}^{k}(i+1)\mu_i.$$

Replacing the roles of i and j in the above inequality and combining it with the equation if the Decomposition Theorem for $\|d\phi\|^2$ yields the needed inequality. ◀

A consequence of this corollary is the following mixing results for λ local spectral expanders:

▶ **Theorem 20** (Mixing of the random walks)**.** *Let X be a weighted pure n-dimensional simplicial complex and let $0 \leq \lambda \leq 1$ be some constant.*

1. *If X is a one-sided λ-local spectral expander, then for every $0 \leq k \leq n-1$,*

$$\forall \phi \in C_0^k(X,\mathbb{R}), \|M_k^+\phi\| \leq \left(\frac{k+1}{k+2} + (k+1)\lambda\right)\|\phi\|.$$

2. *If X is a two-sided λ-local spectral expander, then for every $0 \leq k \leq n-1$,*

$$\forall \phi \in C_0^k(X,\mathbb{R}), \|(M')_k^+\phi\| \leq \left(\frac{k}{k+1} + (k+1)\lambda\right)\|\phi\|.$$

Proof.

1. Let $0 \leq k \leq n-1$ and $\phi \in C_0^k(X,\mathbb{R})$. Assume that X is a one-sided λ-local spectral expander, then by Corollary 19 we get

$$\|d\phi\|^2 \leq \sum_{i=0}^{k}(k+1-i+\sum_{j=i}^{k}(j+1)\mu_j)\|\phi^i\|^2 \leq \sum_{i=0}^{k}(k+1+\sum_{j=0}^{k}(k+1)\lambda)\|\phi^i\|^2 = \quad (3)$$

$$\sum_{i=0}^{k}(k+1+(k+1)^2\lambda)\|\phi^i\|^2 = (k+1+(k+1)^2\lambda)\|\phi\|^2. \quad (4)$$

   Recall that by corollary 11 $(d_k)^* d_k = (k+2)M_k^+$ and therefore

$$\langle M_k^+\phi,\phi\rangle = \frac{1}{k+2}\|d\phi\|^2 \leq (\frac{k+1}{k+2}+\frac{(k+1)^2}{k+2}\lambda)\|\phi\|^2 \leq (\frac{k+1}{k+2}+(k+1)\lambda)\|\phi\|^2.$$

   M_k^+ is a positive operator that maps $C_0^k(X,\mathbb{R})$ into itself and therefore, by the above inequality, any eigenvector of M_k^+ in $C_0^k(X,\mathbb{R})$ has an eigenvalue $\leq \frac{k+1}{k+2}+(k+1)\lambda$ so we are done.

2. Let $0 \leq k \leq n-1$ and $\phi \in C_0^k(X,\mathbb{R})$. Assume that X is a two-sided λ-local spectral expander. By (3), we have that $\|d\phi\|^2 - \|\phi\|^2 \leq (k+(k+1)^2\lambda)\|\phi\|^2$. Also, by Corollary 19, we have that

$$\|d\phi\|^2 \geq \sum_{i=0}^{k}(k+1-i+\sum_{j=i}^{k}(j+1)\nu_j)\|\phi^i\|^2 \geq \sum_{i=0}^{k}(k+1-k+\sum_{j=0}^{k}(k+1)(-\lambda))\|\phi^i\|^2$$

$$\geq \sum_{i=0}^{k}(1-(k+1)^2\lambda)\|\phi^i\|^2 = (1-(k+1)^2\lambda)\|\phi\|^2.$$

   Thus, $\|d\phi\| - \|\phi\|^2 \geq -(k+1)^2\lambda\|\phi\|^2$. Note that

$$\|d\phi\| - \|\phi\|^2 = \langle((k+2)M_+^k - I)\phi,\phi\rangle = (k+1)\langle(M')_+^k\phi,\phi\rangle.$$

   Therefore, after dividing by $(k+1)$ we showed that

$$-(k+1)\lambda\|\phi\|^2 \leq \langle(M')_+^k\phi,\phi\rangle \leq (\frac{k}{k+1}+(k+1)\lambda)\|\phi\|^2.$$

   $(M')_+^k$ is an operator with real eigenvalues that maps $C_0^k(X,\mathbb{R})$ into itself and therefore, by the above inequality, any eigenvector of M_k^+ in $C_0^k(X,\mathbb{R})$ has an eigenvalue between $\frac{k}{k+1}+(k+1)\lambda$ and $-(k+1)\lambda$ so we are done. ◀

Another result of this flavour is mixing theorem for the non-lazy upper random walk in which the condition of the two-sided spectral gap is replaced by the the condition of the one-sided spectral gap and the condition $n >> k$:

▶ **Theorem 21.** *Let X be a weighted pure n-dimensional simplicial complex and let $0 \leq \lambda \leq 1$ be some constant. If X is a one-sided λ-local spectral expander, then for every $0 \leq k \leq n-1$,*

$$\forall\phi \in C_0^k(X,\mathbb{R}), \|(M')_k^+\phi\| \leq \left(\max\left\{\frac{k}{k+1}+(k+1)\lambda, \frac{2(k+1)}{2(n-k-1)-1}\right\}\right)\|\phi\|.$$

Proof. By Lemma 16, for every $0 \leq i \leq k$, $\nu_i \geq -\frac{1}{n-k}$ and by repeating the same argument as in the proof of the previous theorem completes the proof. ◀

References

1 W. Ballmann and J. Świątkowski. On L^2-cohomology and property (T) for automorphism groups of polyhedral cell complexes. *Geom. Funct. Anal.*, 7(4):615–645, 1997. doi:10.1007/s000390050022.

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.

3 Irit Dinur and Tali Kaufman. High dimensional expanders imply agreement expanders. *Electronic Colloquium on Computational Complexity (ECCC)*, 2017.

4 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.

5 Irit Dinur, Subhash Khot, Guy Kindler, Dor Minzer, and Muli Safra. On non-optimally expanding sets in grassmann graphs. *Electronic Colloquium on Computational Complexity (ECCC)*, 24:94, 2017. URL: https://eccc.weizmann.ac.il/report/2017/094.

6 Howard Garland. p-adic curvature and the cohomology of discrete subgroups of p-adic groups. *Ann. of Math. (2)*, 97:375–423, 1973.

7 Anna Gundert and Uli Wagner. On Laplacians of random complexes. In *Computational geometry (SCG'12)*, pages 151–160. ACM, New York, 2012. doi:10.1145/2261250.2261272.

8 Tali Kaufman and David Mass. High dimensional combinatorial random walks and colorful expansion. In *ITCS*, 2017.

9 Alexander Lubotzky, Beth Samuels, and Uzi Vishne. Explicit constructions of ramanujan complexes of type. *Eur. J. Comb.*, 26(6):965–993, 2005. doi:10.1016/j.ejc.2004.06.007.

10 Izhar Oppenheim. Vanishing of cohomology and property (T) for groups acting on weighted simplicial complexes. *Groups Geom. Dyn.*, 9(1):67–101, 2015. doi:10.4171/GGD/306.

11 Izhar Oppenheim. Local spectral expansion approach to high dimensional expanders Part I: Descent of spectral gaps. *Discrete Comput. Geom.*, 59(2):293–330, 2018. doi:10.1007/s00454-017-9948-x.

A Proofs of localization results

Proof of Proposition 12:

Proof. Let $\phi \in C^l(X, \mathbb{R})$, then

$$\sum_{\tau \in X(k)} \|\phi_\tau\|^2 = \sum_{\tau \in X(k)} \sum_{\eta \in X_\tau^{(l-k-1)}} m_\tau(\eta)\phi_\tau(\eta)^2 =$$

$$\sum_{\tau \in X(k)} \sum_{\eta \in X_\tau^{(l-k-1)}} m(\tau \cup \eta)\phi(\tau \cup \eta)^2 = \sum_{\tau \in X(k)} \sum_{\sigma \in X(l), \tau \subset \sigma} m(\sigma)\phi(\sigma)^2 =$$

$$\sum_{\sigma \in X(l)} \sum_{\tau \in X(k), \tau \subset \sigma} m(\sigma)\phi(\sigma)^2 = \binom{l+1}{k+1} \sum_{\sigma \in X(l)} m(\sigma)\phi(\sigma)^2 = \binom{l+1}{k+1} \|\phi\|^2.$$

In order to prove the second equality, we notice that for every $\tau \in X(k)$ and every $\eta \in X_\tau(l-k-2)$, we have that

$$(d^*\phi)_\tau(\eta) = d^*\phi(\tau \cup \eta) = \sum_{\sigma \in X(l), \tau \cup \eta \subset \sigma} \frac{m(\sigma)}{m(\tau \cup \eta)} \phi(\sigma) =$$

$$\sum_{\sigma \setminus \tau \in X(l-k-1), \eta \subset \sigma \setminus \tau} \frac{m_\tau(\sigma \setminus \tau)}{m_\tau(\eta)} \phi_\tau(\sigma \setminus \tau) = d_\tau^* \phi_\tau(\eta).$$

Therefore, $(d^*\phi)_\tau = d^*_\tau \phi_\tau$ and by the first equality of this proposition

$$\binom{l}{k+1}\|d^*\phi\|^2 = \sum_{\tau\in X(k)} \|(d^*\phi)_\tau\|^2 = \sum_{\tau\in X(k)} \|d^*_\tau\phi_\tau\|^2.$$

Assume now that $l < n$, then for every $\sigma \in X(l+1)$, the following holds:

$$(d\phi(\sigma))^2 = (\sum_{\eta\in X(l),\eta\subset\sigma} \phi(\eta))^2 = \sum_{\eta\in X(l),\eta\subset\sigma} \phi(\eta)^2 + \sum_{\eta,\eta'\in X(l),\eta\neq\eta',\eta,\eta'\subset\sigma} 2\phi(\eta)\phi(\eta') =$$
$$\sum_{\eta,\eta'\in X(l),\eta\neq\eta',\eta,\eta'\subset\sigma} (\phi(\eta)+\phi(\eta'))^2 - l\sum_{\eta\in X(l),\eta\subset\sigma}\phi(\eta)^2 =$$
$$\sum_{\tau\in X(l-1),\tau\subset\sigma} (d_\tau\phi_\tau(\sigma\setminus\tau))^2 - l\sum_{\eta\in X(l),\eta\subset\sigma}\phi(\eta)^2.$$

Therefore

$$\|d\phi\|^2 = \sum_{\sigma\in X(l+1)} m(\sigma)(d\phi(\sigma))^2 =$$
$$\sum_{\sigma\in X(l+1)} m(\sigma) \sum_{\tau\in X(l-1),\tau\subset\sigma} (d_\tau\phi_\tau(\sigma\setminus\tau))^2 - l\sum_{\sigma\in X(l+1)} m(\sigma)\sum_{\eta\in X(l),\eta\subset\sigma}\phi(\eta)^2 =$$
$$\sum_{\tau\in X(l-1)}\sum_{\sigma\in X(l+1),\tau\subset\sigma} m(\sigma)(d_\tau\phi_\tau(\sigma\setminus\tau))^2 - l\sum_{\eta\in X(l)}\phi(\eta)^2\sum_{\sigma\in X(l+1),\eta\subset\sigma} m(\sigma) =$$
$$\sum_{\tau\in X(l-1)}\sum_{\gamma\in X^{(1)}_\tau} m_\tau(\gamma)(d_\tau\phi_\tau(\gamma))^2 - l\sum_{\eta\in X(l)} m(\eta)\phi(\eta)^2 = \sum_{\tau\in X(l-1)}\|d_\tau\phi_\tau\|^2 - l\|\phi\|^2 =$$
$$\sum_{\tau\in X(l-1)}\left(\|d_\tau\phi_\tau\|^2 - \frac{l}{l+1}\|\phi_\tau\|^2\right),$$

where the last equality is due to the equality

$$\|\phi\|^2 = \frac{1}{l+1}\sum_{\tau\in X(l-1)}\|\phi_\tau\|^2,$$

proven above. ◀

Proof of Proposition 13:

Proof. Let $\phi \in C^k(X,\mathbb{R})$. Note that for every $\tau \in X(k-1)$, $M^-_{\tau,0}$ is the orthogonal projection on the space of constant functions in $C^0(X_\tau,\mathbb{R})$ and therefore $(M')^+_{\tau,0}M^-_{\tau,0} = M^-_{\tau,0}$.

Further note that by Corollary 11

$$\|d_\tau\phi_\tau\|^2 = \langle 2M^+_{\tau,0}\phi_\tau,\phi_\tau\rangle = \langle((M')^+_{\tau,0}+I)\phi_\tau,\phi_\tau\rangle =$$
$$\langle(M')^+_{\tau,0}\phi_\tau,\phi_\tau\rangle + \|\phi_\tau\|^2 = \langle(M')^+_{\tau,0}M^-_{\tau,0}\phi_\tau,\phi_\tau\rangle + \langle(M')^+_{\tau,0}(I-M^-_{\tau,0})\phi_\tau,\phi_\tau\rangle + \|\phi_\tau\|^2 =$$
$$\|M^-_{\tau,0}\phi_\tau\|^2 + \langle(M')^+_{\tau,0}(I-M^-_{\tau,0})\phi_\tau,\phi_\tau\rangle + \|\phi_\tau\|^2.$$

Therefore, for every $\tau \in X(k-1)$,

$$\|d_\tau\phi_\tau\|^2 - \frac{k}{k+1}\|\phi_\tau\|^2 = \|d_\tau\phi_\tau\|^2 - \|\phi_\tau\|^2 + \frac{1}{k+1}\|\phi_\tau\|^2 =$$
$$\|M^-_{\tau,0}\phi_\tau\|^2 + \langle(M')^+_{\tau,0}(I-M^-_{\tau,0})\phi_\tau,\phi_\tau\rangle + \frac{1}{k+1}\|\phi_\tau\|^2.$$

Also,

$$\|M^-_{\tau,0}\phi_\tau\|^2 = \sum_{v\in X^{(0)}_\tau} m_\tau(\{v\}) \left(\sum_{u\in X^{(0)}_\tau} \frac{m_\tau(\{u\})}{m_\tau(\emptyset)} \phi_\tau(\{u\}) \right)^2 =$$
$$m_\tau(\emptyset) \left(\sum_{u\in X^{(0)}_\tau} \frac{m_\tau(\{u\})}{m_\tau(\emptyset)} \phi_\tau(\{u\}) \right)^2 = \|d^*_\tau \phi_\tau\|^2.$$

Combining this with the previous inequality yields that

$$\|d_\tau\phi_\tau\|^2 - \frac{k}{k+1}\|\phi_\tau\|^2 = \|d^*_\tau\phi_\tau\|^2 + \langle (M')^+_{\tau,0}(I - M^-_{\tau,0})\phi_\tau, \phi_\tau\rangle + \frac{1}{k+1}\|\phi_\tau\|^2.$$

By Proposition 12

$$\|d\phi\|^2 = \sum_{\tau\in X(k-1)} \left(\|d_\tau\phi_\tau\|^2 - \frac{k}{k+1}\|\phi_\tau\|^2 \right),$$

therefore

$$\|d\phi\|^2 = \sum_{\tau\in X(k-1)} \left(\|d^*_\tau\phi_\tau\|^2 + \langle (M')^+_{\tau,0}(I - M^-_{\tau,0})\phi_\tau, \phi_\tau\rangle + \frac{1}{k+1}\|\phi_\tau\|^2 \right).$$

Using the equalities proven in Proposition 12, we deduce that

$$\|d\phi\|^2 = \|d^*\phi\|^2 + \|\phi\|^2 + \sum_{\tau\in X(k-1)} \langle (M')^+_{\tau,0}(I - M^-_{\tau,0})\phi_\tau, \phi_\tau\rangle,$$

as needed. ◀

Proof of Lemma 14:

Proof. Let ϕ be as above. Recall that for every $\tau \in X(k-1)$, ϕ_τ decomposes orthogonally as

$$\phi_\tau = (I - M^-_{\tau,0})\phi_\tau + M^-_{\tau,0}\phi_\tau,$$

Therefore

$$\langle (M')^+_{\tau,0}(I - M^-_{\tau,0})\phi_\tau, \phi_\tau\rangle =$$
$$\langle (M')^+_{\tau,0}(I - M^-_{\tau,0})\phi_\tau, (I - M^-_{\tau,0})\phi_\tau\rangle + \langle (M')^+_{\tau,0}(I - M^-_{\tau,0})\phi_\tau, M^-_{\tau,0}\phi_\tau\rangle.$$

As explained above, $(M')^+_{\tau,0}(I - M^-_{\tau,0})\phi_\tau \in \mathrm{Im}(I - M^-_{\tau,0})$ and therefore

$$\langle (M')^+_{\tau,0}(I - M^-_{\tau,0})\phi_\tau, M^-_{\tau,0}\phi_\tau\rangle = 0.$$

This yields that

$$\langle (M')^+_{\tau,0}(I - M^-_{\tau,0})\phi_\tau, \phi_\tau\rangle = \langle (M')^+_{\tau,0}(I - M^-_{\tau,0})\phi_\tau, (I - M^-_{\tau,0})\phi_\tau\rangle.$$

Note that by definitions of μ_k, ν_k

$$\langle (M')^+_{\tau,0}(I - M^-_{\tau,0})\phi_\tau, (I - M^-_{\tau,0})\phi_\tau\rangle \le \mu_k \|(I - M^-_{\tau,0})\phi_\tau\|^2 \le \mu_k\|\phi_\tau\|^2,$$

and

$$\langle (M')^{+}_{\tau,0}(I - M^{-}_{\tau,0})\phi_\tau, (I - M^{-}_{\tau,0})\phi_\tau \rangle \geq \nu_k \|(I - M^{-}_{\tau,0})\phi_\tau\|^2 \geq \nu_k \|\phi_\tau\|^2.$$

Summing over all $\tau \in X(k-1)$ and applying Proposition 12 yields the needed results, i.e.,

$$\sum_{\tau \in X(k-1)} \langle (M')^{+}_{\tau,0}(I - M^{-}_{\tau,0})\phi_\tau, \phi_\tau \rangle =$$

$$\sum_{\tau \in X(k-1)} \langle (M')^{+}_{\tau,0}(I - M^{-}_{\tau,0})\phi_\tau, (I - M^{-}_{\tau,0})\phi_\tau \rangle \leq \sum_{\tau \in X(k-1)} \mu_k \|\phi_\tau\|^2 = (k+1)\mu_k \|\phi\|^2,$$

and a similar computation yields the second inequality of the Lemma. ◀

Improved Composition Theorems for Functions and Relations

Sajin Koroth[1]
Department of Computer Science, University of Haifa, Haifa 3498838, Israel
sajin@csweb.haifa.ac.il
https://orcid.org/0000-0002-7989-1963

Or Meir[2]
Department of Computer Science, University of Haifa, Haifa 3498838,Israel
ormeir@cs.haifa.ac.il
https://orcid.org/0000-0001-5031-0750

Abstract

One of the central problems in complexity theory is to prove super-logarithmic depth bounds for circuits computing a problem in P, i.e., to prove that P is not contained in NC^1. As an approach for this question, Karchmer, Raz and Wigderson [5] proposed a conjecture called the KRW conjecture, which if true, would imply that P is not cotained in NC^1.

Since proving this conjecture is currently considered an extremely difficult problem, previous works by Edmonds, Impagliazzo, Rudich and Sgall [1], Håstad and Wigderson [3] and Gavinsky, Meir, Weinstein and Wigderson [2] considered weaker variants of the conjecture. In this work we significantly improve the parameters in these variants, achieving almost tight lower bounds.

2012 ACM Subject Classification Theory of computation → Communication complexity, Theory of computation → Circuit complexity, Theory of computation → Complexity classes

Keywords and phrases circuit complexity, communication complexity, KRW conjecture, composition

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.48

1 Introduction

The holy grail question in circuit complexity is to prove super-polynomial size lower bounds for **NP**, i.e., to show that **NP** is not contained in **P**/poly. This is considered to be an extremely difficult problem and even after years of research we do not even know a super-linear size lower bound. Hence, a natural approach is to prove lower bounds for more restricted classes of circuits. One such restricted class of circuits are $\mathbf{NC}^1$, which are circuits of polynomial size, logarithmic depth and bounded fan-in. It is widely believed that $\mathbf{NC}^1$ does not contain **P**. However even this problem is deemed very hard. In particular, we do not even know super-linear lower bounds for $\mathbf{NC}^1$ circuits computing a function even in **NEXP**.

An approach for separating **P** from $\mathbf{NC}^1$ was suggested by Karchmer, Raz and Wigderson [5]. They conjectured that the depth complexity of Boolean functions adds up under a certain composition of Boolean functions. We refer to this as the KRW conjecture, and if it is true, would give an explicit function in **P** which does not have $\mathbf{NC}^1$ circuits. In order to state the conjecture we define the following composition of functions.

[1] Supported by the Israel Science Foundation (grant No. 1445/16)
[2] Partially supported by the Israel Science Foundation (grant No. 1445/16)

▶ **Definition 1** (Composition)**.** Given two arbitrary Boolean functions $f : \{0,1\}^m \to \{0,1\}$, $g : \{0,1\}^n \to \{0,1\}$ we define their *composition* to be the Boolean function $f \diamond g : (\{0,1\}^n)^m \to \{0,1\}$ obtained as follows:

$$f \diamond g\,(x_1, x_2, \ldots, x_m) \quad = \quad f\,(g\,(x_1)\,, g\,(x_2)\,, \ldots, g\,(x_m))$$

where $x_i \in \{0,1\}^n$,$1 \leq i \leq m$.

We recall a standard definition of *depth complexity*. For a Boolean function $f : \{0,1\}^n \to \{0,1\}$, its depth complexity denoted by $D(f)$, is the smallest depth of a circuit of AND, OR and NOT gates of fan-in 2 that computes f. We now state the KRW conjecture.

▶ **Conjecture 2** (The KRW conjecture, [5])**.** *Let* $f : \{0,1\}^m \to \{0,1\}$ *and* $g : \{0,1\}^n \to \{0,1\}$ *be two arbitrary non-constant Boolean functions. Then the following holds true*[3]*,*

$$D\,(f \diamond g) \quad \approx \quad D\,(f) + D\,(g)$$

Since this is a difficult conjecture previous works [1, 2, 3] proved lower bounds for two simplified variants of the KRW conjecture. In this work we improve the parameters of these lower bounds, motivated by the value of these parameters in the target application. Though the two simplified variants of the KRW conjecture are important milestones, we would like to point out that settling these two variants of the conjecture do not imply any circuit lower bound.

1.1 Background

1.1.1 Karchmer-Wigderson relations

Karchmer and Wigderson [6] established an interesting connection between the depth complexity of a Boolean function f and the communication complexity of an associated relation, which is called the Karchmer-Wigderson relation, and is denoted by KW_f. They proved that the depth complexity of f is equal to the deterministic communication complexity of KW_f.

The Karchmer-Wigderson relation associated with a Boolean function $f : \{0,1\}^n \to \{0,1\}$ is the following communication problem. Alice is given an input $x \in f^{-1}(0)$ and Bob is given an input $y \in f^{-1}(1)$.The objective of the players is to find an index $i \in [n]$ such that $x_i \neq y_i$.

In the rest of this paper we refer to the Karchmer-Wigderson relation as "KW relation". We also denote the deterministic communication complexity of KW_f by $\mathrm{CC}\,(KW_f)$. In the rest of this paper, when we say communication complexity, we mean deterministic communication complexity.

1.1.2 The KW relation of composition

We study the KRW conjecturing using the Karchmer-Wigderson framework. To this end, we describe how the KW relation related to the composition $f \diamond g$ looks like for arbitrary Boolean functions f, g. In the KW relation $KW_{f \diamond g}$, Alice and Bob's inputs are conveniently viewed as $m \times n$ matrices X, Y, respectively. Given an $m \times n$ binary matrix X, we define $g(X)$ to be an m bit binary vector obtained by applying g to the rows of X.

The KW relation $KW_{f \diamond g}$ associated with the composition $f \diamond g$ is the following communication problem.

[3] The approximate equality in the statement of the conjecture is intentionally left vague, since there are multiple possible definitions which are weaker than strict equality, but which would still imply the $\mathbf{P} \not\subseteq \mathbf{NC}^1$ separation.

- Alice gets a matrix $X \in \{0,1\}^{m \times n}$ with the promise that the vector $a = g(X)$ is such that $f(a) = 0$.
- Bob gets a matrix $Y \in \{0,1\}^{m \times n}$ with the promise that the vector $b = g(Y)$ is such that $f(b) = 1$.
- The goal of the players is to find a pair of indices $(i,j) \in [m] \times [n]$ such that $X_{i,j} \neq Y_{i,j}$.

It is easy to see that $CC(KW_{f \diamond g}) \leq \mathrm{CC}(KW_f) + \mathrm{CC}(KW_g)$. The KRW conjecture says that the upper bound is essentially optimal for $KW_{f \diamond g}$.

1.1.3 Universal relation and its composition

Because of the difficulty of the original conjecture, Karchmer, Raz and Wigderson [5] suggested studying a simpler variant of the conjecture. To this end, they defined a simplified variant of KW relations called the universal relation. The *universal relation* on n bits, denoted by U_n, is a promise problem where Alice is given an input $x \in \{0,1\}^n$ and Bob is given an input $y \in \{0,1\}^n$ with the guarantee that $x \neq y$. The goal of the players is to find an index $i \in [n]$ such that $x_i \neq y_i$. This relation is called the universal relation in the sense that KW relation of any Boolean function reduces to it. Tardos and Zwick [9] proved that deterministic communication complexity of U_n is $n+1$.

Karchmer, Raz and Wigderson [5] suggested to study the composition of universal relations defined next. The composition of the universal relation U_m with the universal relation U_n, denoted by $U_m \diamond U_n$, is defined as the following communication problem.

- Alice gets a matrix $X \in \{0,1\}^{m \times n}$ and a vector $a \in \{0,1\}^m$.
- Bob gets a matrix $Y \in \{0,1\}^{m \times n}$ and a vector $b \in \{0,1\}^m$.
- They are guaranteed that $a \neq b$.
- They are guaranteed that for any $i \in [m]$, whenever $a_i \neq b_i$, the corresponding rows of the matrices, denoted by X_i and Y_i, are not equal.
- The goal of the players is to find a pair of indices $(i,j) \in [m] \times [n]$ such that $X_{i,j} \neq Y_{i,j}$.

This problem generalizes the KW relation corresponding to the composition of Boolean functions. As a first step towards the KRW conjecture, [5] suggested to prove the following:

▶ **Conjecture 3** (Analogue of KRW conjecture for $U_m \diamond U_n$, [5]). $CC(U_m \diamond U_n) \approx CC(U_m) + CC(U_n) \approx m + n$

The first progress towards this conjecture on the composition of universal relations was made by Edmonds et al. [1] who proved that $\mathrm{CC}(U_n \diamond U_n) \geq 2n - O(\sqrt{n})$. Later Håstad and Wigderson [3] improved on the results of [1] using a completely different proof strategy. They [3] proved that for sufficiently large n, $\mathrm{CC}(U_n \diamond U_n) \geq 2n - 1$.

1.1.4 Composition of a function with the universal relation

A variant of the above conjecture, which is closer to the original KRW conjecture, is a conjecture dealing with composition of an arbitrary Boolean function with a universal relation. It was suggested by Gavinsky et al. [2]. Given an arbitrary Boolean function $f : \{0,1\}^m \to \{0,1\}$, they define the KW relation $KW_{f \diamond U_n}$ as the following promise problem.

- Alice gets a matrix $X \in \{0,1\}^{m \times n}$ and a vector $a \in \{0,1\}^m$ such that $f(a) = 0$.
- Bob gets a matrix $Y \in \{0,1\}^{m \times n}$ and a vector $b \in \{0,1\}^m$ such that $f(b) = 1$.
- They are guaranteed that for any $i \in [m]$ whenever $a_i \neq b_i$, the corresponding rows of the matrices, denoted by X_i and Y_i, are not equal.
- The goal of the players is to find a pair of indices $(i,j) \in [m] \times [n]$ such that $X_{i,j} \neq Y_{i,j}$.

This definition is a natural candidate for the composition of a function with the universal relation as any instance of $KW_{f \diamond U_n}$ as defined above is also a legal instance of $U_m \diamond U_n$. In addition, any instance of $KW_{f \diamond g}$ where $f : \{0,1\}^m \to \{0,1\}$ and $g : \{0,1\}^n \to \{0,1\}$ is also a legal instance of $KW_{f \diamond U_n}$ if we set $a = g(X)$ and $b = g(Y)$.

We now describe the definition of the promise version of the composition of a function f with a universal relation, due to [2]. As in the above version, goal of Alice and Bob is to find an entry (i, j) on which X and Y differ, but they are allowed to reject if there exists an index $i \in [m]$ such that $a_i \neq b_i$ but $X_i = Y_i$.

▶ Remark. It is easy to see that the complexity of the problem over all inputs is larger by at most two bits than the complexity of the promise problem.

Gavinsky et al. [2] proposed the following analogous conjecture for $KW_{f \diamond U_n}$:

▶ **Conjecture 4** (Conjecture for $f \diamond U_n$, [2]). $CC(KW_{f \diamond U_n}) \approx CC(KW_f) + CC(U_n) \approx CC(KW_f) + n$

They [2] also proved the following lower bound on the composition[4].

▶ **Theorem 5** (Lower bound for $f \diamond U_n$, [2]). *For any $m, n \in \mathbb{N}$, and any non-constant function $f : \{0,1\}^m \to \{0,1\}$, $CC(KW_{f \diamond U_n}) \geq \log(L(f)) + n - O\left(1 + \frac{m}{n}\right) \log m$.*

1.1.5 Known results

In this work, we are interested in improving the dependence on m in the known lower bounds, and in particular in the setting where $m \gg n$. The early results of Edmonds et al. [1] and Håstad and Wigderson [3], established lower bounds for the composition of universal relation with itself when $m = n$. However, these results can be generalized in a straightforward manner for proving lower bounds for communication complexity of $U_m \diamond U_n$.

A straightforward generalization of the [1] bound for $m \neq n$ yields the following result: $\mathrm{CC}(U_m \diamond U_n) \geq m + n - O(\sqrt{m})$

A straightforward generalization of the Håstad and Wigderson [3] result only gives a lower bound of $2n - o(1)$ for $\mathrm{CC}(U_m \diamond U_n)$. Note that this lower bound is independent of m and hence is far from the conjectured lower bound of $m + n$ when $m \gg n$.

Gavinsky et al. studied $\mathrm{CC}(f \diamond U_n)$ and proved a lower bound of $\log(L(f)) + n - O\left(1 + \frac{m}{n}\right) \log m$. The lower bound proved by [2] is for m and n which are not necessarily equal. However, similar to the other known lower bounds it also has a loss term depending on m, $O\left(\frac{m}{n} \log m\right)$, which becomes significant if $m \gg n$.

1.2 Our results

We overcome the additive losses in the above lower bounds and obtain bounds which are optimal except for an $O(\log^\star m)$ additive term.

▶ **Theorem 6.** *For any $m, n \in \mathbb{N}$ with $n \geq 6 \log m$, and any non-constant function $f : \{0,1\}^m \to \{0,1\}$, $CC(KW_{f \diamond U_n}) \geq \log L(f) + n - O(\log^\star m)$ where $L(f)$ is the formula complexity of the function f.*

[4] Note that the lower bound depends on the logarithm of the formula complexity of f, denoted by $\log(L(f))$, instead of the depth complexity of f denoted by $D(f)$. However, the parameters $\log(L(f))$ and $D(f)$ are tightly related.

We also prove a similar lower bound for the deterministic communication complexity of $U_m \diamond U_n$.

▶ **Theorem 7.** *For any $m, n \in \mathbb{N}$ with $n \geq 6 \log m$, $CC(U_m \diamond U_n) \geq m + n - O(\log^\star m)$*

But for sake of simplicity, we prove the following lower bounds with $O(\log m)$ additive losses and defer the proof of $O(\log^\star m)$ improvement to the full version of the paper.

▶ **Theorem 8.** *For any $m, n \in \mathbb{N}$, and any non-constant function $f : \{0,1\}^m \to \{0,1\}$, $CC(KW_{f \diamond U_n}) \geq \log L(f) + n - O(\log m)$ where $L(f)$ is the formula complexity of the function f.*

As a corollary we get the following lower bound for the deterministic communication complexity of $U_m \diamond U_n$.

▶ **Corollary 9.** *For any $m, n \in \mathbb{N}$, $CC(U_m \diamond U_n) \geq m + n - O(\log m)$*

Proof. As noted earlier, for any non-constant Boolean function $f : \{0,1\}^m \to \{0,1\}$, the KW relation associated with $f \diamond U_n$ reduces to the communication problem $U_m \diamond U_n$. In particular, this implies that for any non-constant $f : \{0,1\}^m \to \{0,1\}$, it holds that $\mathrm{CC}(U_m \diamond U_n) \geq \mathrm{CC}(KW_{f \diamond U_n})$. Since there exists Boolean functions with formula complexity $\frac{2^m}{\log m}$ among the set of all m-bit Boolean functions, we get by Theorem 8 that, $\mathrm{CC}(U_m \diamond U_n) \geq \log\left(\frac{2^m}{\log m}\right) + n - O(\log m) = m + n - O(\log m)$. ◀

Note that in the improved lower bounds, since the losses are only $O(\log^\star m)$, we do not get the improved lower bound for $U_m \diamond U_n$ as a corollary from the lower bound for $f \diamond U_n$. But using a very similar proof, an independent lower bound can be derived for $U_m \diamond U_n$.

Motivation

Bridging the gap between the known lower bounds and the conjectured lower bounds is an important pursuit in itself. More importantly, the KRW conjecture implies the two weaker conjectures up to sub-logarithmic additive losses. Thus it is necessary to prove the weaker variants with such losses if one it to settle the KRW conjecture. Our work obtains the first known lower bounds on the two variants of the conjecture, with additive losses smaller than the losses implied if the KRW conjecture is true.

However, the improved parameters in our lower bounds are also significant for the following reason. As we mentioned above, the main motivation for studying the KRW conjecture is that it implies $\mathbf{P} \not\subseteq \mathbf{NC}^1$. In fact, in order to obtain this implication, it suffices to prove a relaxed variant of the conjecture in which g is a random function (but f is an arbitrary function). This relaxation seems to be closer to our reach, since [2] have already proved the conjecture when g is replaced with the universal relation and f is an arbitrary function.

However, in order to derive $\mathbf{P} \not\subseteq \mathbf{NC}^1$ from the relaxation, we need to have lower bounds that are meaningful for values of m that satisfy $m = n^{\omega(1)}$. Unfortunately, the result of [2] does not give a meaningful bound when $m \geq n^2$. If we aspire to prove this relaxation of the KRW conjecture, we need to have lower bounds that work for larger values of m. Our Theorem 8 achieves exactly that: it gives a bound on $KW_{f \diamond U_n}$ that is meaningful as long as $m = o(2^n)$, which is good enough for our purposes.

We should mention that there is another relaxation of the KRW conjecture that implies $\mathbf{P} \neq \mathbf{NC}^1$, in which f is a random function and g is an arbitrary function. However, this relaxation seems farther from our reach, since we do not have any result on $KW_{U_m \diamond g}$ when g is an arbitrary function.

Results for formula lower bounds

We also have a similar (though somewhat weaker) result for formula lower bounds. The analogue of the KRW conjecture for formula lower bounds says that $L(f \diamond g) \approx L(f) \cdot L(g)$ (see [2] for details). This can also be stated in the Karchmer-Wigderson framework: To this end, for any communication problem R, let us denote by $L(R)$ the smallest number of distinct transcripts in a deterministic protocol that solves R. Then, the analogue of the KRW conjecture is to say that $L(KW_{f \diamond g}) \approx L(KW_f) \cdot L(KW_g)$ or equivalently $\log L(KW_{f \diamond g}) \approx \log L(KW_f) + \log L(KW_g)$ We prove that $\log L(KW_{f \diamond U_n}) \geq \log L(KW_f) + n - O\left(\sqrt{\log L(KW_f)} + \log m\right)$. While this result is weaker than our result for depth complexity due to the loss of $O(\sqrt{\log L(KW_f)})$, it is still stronger than the corresponding results of [3, 2] when $\log L(KW_f) \gg n$[5]. We defer the proof to the full version of this paper.

1.3 Our techniques

In this section we provide an overview of the proof of our main result (Theorem 8). Our approach is based on a proof strategy due to [1]. We begin by describing this proof strategy, and then describe our new ideas. Recall that the inputs of the parties are pairs (X, a) and (Y, b) where X, Y are $m \times n$ matrices and $a \in f^{-1}(0), b \in f^{-1}(1)$ are m bit column vectors. Fix a deterministic protocol Π that solves $KW_{f \diamond U_n}$. We wish to prove that Π must transmit $\approx \mathrm{CC}(KW_f) + n$ bits.

The basic intuition that underlies the proof consists of the following three observations:

- Morally, in order to solve $KW_{f \diamond U_n}$, the parties have to solve the universal relation on one of the rows of X, Y. To this end, they have to transmit at least n bits about this row.
- However, solving the universal relation on a row X_i, Y_i only makes sense if the parties are guaranteed that $X_i \neq Y_i$, since otherwise they might waste their communication on equal rows.
- Intuitively, in order to find rows X_i, Y_i that are guaranteed to be different, the parties must solve KW_f, and to this end they have to transmit at least $\mathrm{CC}(KW_f)$ bits.

Therefore, the total amount of communication must be at least $\approx \mathrm{CC}(KW_f) + n$.

High-level idea

We now sketch the argument that is based on this intuition. We partition the communication of the protocol Π into two stages – intuitively, the parties should solve KW_f on a, b in the first stage, and then solve the universal relation on a row of X, Y in the second stage. Formally, the first stage is defined as the first $\mathrm{CC}(KW_f) - \alpha$ bits that are transmitted in the protocol, where α is some small "slack term", and the second stage consists of the remaining bits. We prove that the parties must transmit approximately n bits during the second stage, and this implies that the protocol must transmit approximately $\mathrm{CC}(KW_f) + n$ bits in total.

We start by making the following observation: If, during the first stage, the parties only "talk about" a, b, then it is easy to prove that they have to transmit at least n bits in the second stage: morally, this is because in the second stage they still have to solve the universal relation on one of the rows of X, Y, and must do so "from scratch" (since they did not talk about this row before).

[5] the work of [1] does not provide a result for formula lower bounds

The challenging case is when the parties "talk about" X, Y during the first stage. We deal with this case by observing that talking about X, Y during the first stage is useless, for the following reason: during the first stage, the parties have not finished solving KW_f yet, since they communicated less than $\mathrm{CC}(KW_f)$ bits. Thus, at this point, they do not know any row i where $a_i \neq b_i$, and therefore they do not know any row i for which they are guaranteed that $X_i \neq Y_i$. This means that any communication about X, Y during the first stage is likely to be wasted on rows where $X_i = Y_i$, and hence will not help the parties to solve the problem.

In short, we observe that any communication about X, Y during the first stage is useless. Therefore, any optimal protocol should behave roughly as if the parties do not talk about X, Y at all during the first stage. However, in such case the parties must transmit n bits during the second stage, and this is what we want to prove.

An adversary argument

The foregoing idea is formalized using an adversary argument. This means that we view the inputs of the parties as if they are chosen by an adversary that can adapt to the messages sent by the parties. We now describe the behavior of our adversary. The adversary starts by letting the parties talk throughout the first stage, and chooses the messages such that no more than $\mathrm{CC}(KW_f) - \alpha$ bits of information are revealed. At the end of the first stage, the adversary looks at the transcript of the communication so far, and partitions the rows of X, Y into two types:

- "Revealed rows", about which the parties talked much (i.e., more than τ bits for some small parameter τ).
- "Unrevealed rows", about which the parties talked a little (i.e., at most τ bits).

Intuitively, if the parties end up solving the universal relation on one of the unrevealed rows, then they have to transmit about $n - \tau \approx n$ bits in the second stage, which is what we want to prove.

Hence, the adversary only needs to worry about the revealed rows. In order to deal with the revealed rows, the adversary chooses the inputs of the parties such that $a_i = b_i$ for every revealed row X_i, Y_i. This allows the adversary to prevent the parties from solving the universal relation on a revealed row X_i, Y_i, since the adversary is free to set $X_i = Y_i$ whenever they attempt to do so. It follows that the parties must solve the universal relation on an unrevealed row, and therefore they must transmit roughly n bits during the second stage, as required.

In order for this argument to go through, we must make sure that the adversary is indeed capable of setting $a_i = b_i$ for every revealed row. In principle, the adversary should be able to set $a_i = b_i$ for some rows because the parties have not yet finished solving KW_f. However, the *number* of rows for which the adversary can set $a_i = b_i$ depends on how far the parties are from solving KW_f exactly. Morally, it can be shown that if the parties talked at most $\mathrm{CC}(KW_f) - k$ bits about a and b, then the adversary can set $a_i = b_i$ for about k rows. Therefore, in order for the argument to work, the number of revealed rows should be at most k. This condition depends, in turn, on the choice of the threshold τ (which determines which rows are considered "revealed") and on the choice of the slack term α (which determines the length of the first stage). This is our point of departure from the work of [1].

The analysis of [1]

The analysis of [1] sets both the threshold τ and the slack term α to[6] $O(\sqrt{\mathrm{CC}(KW_f)})$ (so the length of the first stage is $\mathrm{CC}(KW_f) - O(\sqrt{\mathrm{CC}(KW_f)})$ bits). The analysis proceeds as follows: In the first stage, the parties talked at most $\mathrm{CC}(KW_f)$ bits about the matrices X, Y (since they talked at most $\mathrm{CC}(KW_f) - O(\sqrt{\mathrm{CC}(KW_f)})$ bits overall). By Markov's inequality, this implies that the number of revealed rows is at most

$$\frac{\mathrm{CC}(KW_f)}{\tau} = O(\sqrt{\mathrm{CC}(KW_f)}).$$

On the other hand, the parties talked at most $\mathrm{CC}(KW_f) - O(\sqrt{\mathrm{CC}(KW_f)})$ bits about a and b (again, since they talked at most $\mathrm{CC}(KW_f) - O(\sqrt{\mathrm{CC}(KW_f)})$ bits overall), and hence the adversary can set $a_i = b_i$ for $O(\sqrt{\mathrm{CC}(KW_f)})$ rows. Thus, for an appropriate choice of the parameters, the adversary can set $a_i = b_i$ for all the revealed rows.

Note that this analysis loses a term of $O(\sqrt{\mathrm{CC}(KW_f)})$ twice: once because it sets $\alpha = O(\sqrt{\mathrm{CC}(KW_f)})$ (thus losing $O(\sqrt{\mathrm{CC}(KW_f)})$ bits in the first stage), and once because it sets $\tau = O(\sqrt{\mathrm{CC}(KW_f)})$ (thus losing $O(\sqrt{\mathrm{CC}(KW_f)})$ in the second stage).

Our analysis

Our goal is to avoid the loss of the $O(\sqrt{\mathrm{CC}(KW_f)})$ term in the lower bound, and therefore we set $\tau = O(1)$ and $\alpha = O(1)$. In order to make the analysis go through with this choice of parameters, we need a new idea.

Our first key idea is the following observation: the analysis of [1] works as if the parties transmit during the first stage $\mathrm{CC}(KW_f)$ bits about X, Y, and another $\mathrm{CC}(KW_f)$ bits about a, b. However, this cannot happen, since the total amount of communication in the first stage is less than $\mathrm{CC}(KW_f)$. Hence, if the parties talked much about X, Y, then they can talk only a little about a, b, and vice versa.

For concreteness, let us denote by ℓ the number of bits that the parties talked about X, Y. Then, by Markov's inequality, the number of revealed rows is at most $\frac{\ell}{\tau}$. On the other hand, the parties talked at most $\mathrm{CC}(KW_f) - \alpha - \ell$ bits about a and b (since they talked at most $\mathrm{CC}(KW_f) - \alpha$ bits overall). Hence, the adversary can set $a_i = b_i$ for ℓ rows, which is more than the number of revealed rows as long as $\tau > 1$.

A further complication

The foregoing simple idea almost works. However, there is still another complication that we have not discussed: when the adversary fixes $a_i = b_i$ for a row, it may leak a bit of information to the parties, and in total, if there are k revealed rows then k bits may be leaked. This leakage causes us to lose k bits in the lower bound of the second stage: in the worst case, after the leakage the parties know $\tau + k$ bits about some unrevealed row, and therefore, in the second stage, they may solve the universal relation on this row using only $n - \tau - k$ bits. In such case, we will lose a term of k in the lower bound.

In the work of [1], this loss was not an issue, since they had $k = O(\sqrt{\mathrm{CC}(KW_f)})$ revealed rows, and they were losing a term of $O(\sqrt{\mathrm{CC}(KW_f)})$ anyhow. However, in our argument above the number of revealed rows is about $\frac{\ell}{\tau}$, which could be almost as large as $\mathrm{CC}(KW_f)$. This loss is therefore unacceptable.

---

[6] Note that the original paper of [1] proves the result for $U_n \diamond U_n$ rather than $KW_{f \diamond U_n}$, and therefore all the occurrences of $\mathrm{CC}(KW_f)$ in the following description were originally equal to n.

An iterative adversary

Our second key idea is to modify the adversary in order to deal with the above complication as follows: After the adversary fixed $a_i = b_i$ for each of the revealed rows, it checks if new revealed rows were created in the process. In other words, the adversary checks if the leakage of information caused the parties to know more than τ bits on some of the previously unrevealed rows. If this is the case, the adversary fixes $a_i = b_i$ for each of these new revealed rows as well. The adversary now repeats the process until there are no more revealed rows – i.e., until the parties know at most τ bits of information on each of the unrevealed rows.

Obviously, if such an adversary can be implemented, then it avoids losing the term of k in the second stage. However, we need to show that such an adversary can indeed be implemented, i.e., that the adversary is allowed to fix $a_i = b_i$ for all the revealed rows encountered throughout all the iterations.

To this end, we bound the total number of revealed rows that are encountered in the process using the following potential argument: Whenever the adversary fixes $a_i = b_i$ for a revealed row X_i, Y_i, the parties may gain one bit of information that is leaked about X, Y. However, the parties also lose τ bits of information about X, Y, in the sense that the τ bits of information they knew about X_i, Y_i become useless after the fixing. In total, whenever the adversary fixes $a_i = b_i$ for a revealed row, the parties lose $(\tau - 1)$ bits of information about X, Y. Since at the beginning of the iterative process the parties knew at most ℓ bits of information about X, Y, the total number of revealed rows cannot exceed $\ell/(\tau - 1)$.

Finally, recall the adversary is allowed to fix $a_i = b_i$ for ℓ rows, and observe that this is more than $\ell/(\tau - 1)$ as long as $\tau \geq 2$. Therefore, we can implement the above adversary. This concludes the analysis.

1.4 Organization of the paper

In Section 2 we review the required preliminaries. In Section 3 we discuss the general adversarial strategy and prove a lower bound for the composition of a function with the universal relation.

2 Preliminaries

We use $[n]$ to denote the set $\{1, \ldots, n\}$. We denote the set of $m \times n$ binary matrices by $\{0,1\}^{m \times n}$. For every binary $m \times n$ matrix X, we denote by $X_i \in \{0,1\}^n$ the i-th row of X.

2.1 Formulas

In this paper we consider *(de-Morgan) formulas* of whose internal gates are 2 bit, AND ($\wedge$) or OR ($\vee$) gates.

▶ **Definition 10.** The *formula complexity* of a Boolean function $f : \{0,1\}^n \to \{0,1\}$, denoted $L(f)$, is the size of the smallest formula[7] that computes f. The *depth complexity* of f, denoted $D(f)$, is the smallest depth of a formula that computes f.

[7] Note that we define here the depth complexity of a function as the depth of a *formula* that computes f, while in the introduction we defined it as the depth of a *circuit with fan-in* 2 that computes f. However, for our purposes, this distinction does not matter, since every circuit with fan-in 2 can be converted into a formula with the same depth.

The following definition generalizes the above definitions from functions to promise problems, which will be useful when we discuss Karchmer-Wigderson relations.

▶ **Definition 11.** Let $\mathscr{X}, \mathcal{Y} \subseteq \{0,1\}^n$ be disjoint sets. We say that a formula ϕ *separates* $\mathscr{X}$ and $\mathcal{Y}$ if $\phi(\mathscr{X}) = 0$ and $\phi(\mathcal{Y}) = 1$. The *formula complexity of the rectangle* $\mathscr{X} \times \mathcal{Y}$, denoted $L(\mathscr{X} \times \mathcal{Y})$, is the size of the smallest formula that separates $\mathscr{X}$ and $\mathcal{Y}$. The *depth complexity of the rectangle* $\mathscr{X} \times \mathcal{Y}$, denoted $D(\mathscr{X} \times \mathcal{Y})$, is the smallest depth of a formula that separates $\mathscr{X}$ and $\mathcal{Y}$.

2.2 Communication complexity

Let $\mathscr{X}$, $\mathcal{Y}$, and $\mathcal{Z}$ be sets, and let $R \subseteq \mathscr{X} \times \mathcal{Y} \times \mathcal{Z}$ be a relation. The communication problem [10] that corresponds to R is the following: two players, Alice and Bob, get inputs $x \in \mathscr{X}$ and $y \in \mathcal{Y}$, respectively. They would like to communicate and find $z \in \mathcal{Z}$ such that $(x, y, z) \in R$. At each round, one of the players sends a bit that depends on her/his input and on the previous messages, until they find z. The *communication complexity* of R is the minimal number of bits that is transmitted by a protocol that solves R.

We now define a notion of protocol size that is analogous to the notion of formula size.

▶ **Definition 12.** We define the *size* of a protocol Π to be its number of leaves. Note that this is also the number of distinct transcripts of the protocol. We define the *protocol size* of a relation R, denoted $L(R)$, as the size of the smallest protocol that solves it (this is also known as the *protocol partition number* of R).

2.3 Karchmer-Wigderson relations

In this section, we define KW relations formally, and state the correspondence between KW relations and formulas. We start by defining KW relations for general rectangles, and then specialize the definition to functions.

▶ **Definition 13.** Let $\mathscr{X}, \mathcal{Y} \subseteq \{0,1\}^n$ be two disjoint sets. The *KW relation* $KW_{\mathscr{X} \times \mathcal{Y}} \subseteq \mathscr{X} \times \mathcal{Y} \times [n]$ is defined by $KW_{\mathscr{X} \times \mathcal{Y}} = \{(x, y, i) : x_i \neq y_i\}$ Intuitively, $KW_{\mathscr{X} \times \mathcal{Y}}$ corresponds to the communication problem in which Alice gets $x \in \mathscr{X}$, Bob gets $y \in \mathcal{Y}$, and they would like to find a coordinate $i \in [n]$ such that $x_i \neq y_i$ (note that $x \neq y$ since $\mathscr{X} \cap \mathcal{Y} = \emptyset$).

▶ **Definition 14.** Let $f : \{0,1\}^n \to \{0,1\}$ be a non-constant function. The *KW relation of* f, denoted KW_f, is defined by $KW_f = KW_{f^{-1}(0) \times f^{-1}(1)}$.

We now state the connection between Boolean functions and Karchmer-Wigderson relations.

▶ **Theorem 15** ([6][8]). *For every two disjoint sets* $\mathscr{X}, \mathcal{Y} \subseteq \{0,1\}^n$ *it holds that* $D(\mathscr{X} \times \mathcal{Y}) = CC(KW_{\mathscr{X} \times \mathcal{Y}})$*, and* $L(\mathscr{X} \times \mathcal{Y}) = L(KW_{\mathscr{X} \times \mathcal{Y}})$*. In particular, for every non-constant* $f : \{0,1\}^n \to \{0,1\}$*, it holds that* $D(f) = CC(KW_f)$*, and* $L(f) = L(KW_f)$*.*

We note that for a pair of disjoints sets $\mathscr{X}, \mathcal{Y} \subseteq \{0,1\}^n$ where either $\mathscr{X}$ or $\mathcal{Y}$ or both is the empty set, the leaf complexity $L(\mathscr{X} \times \mathcal{Y})$ is defined to 0.

Throughout this work, we will rely extensively on the following *subadditivity property* of protocol size and formula complexity: for every $\mathscr{X}, \mathcal{Y} \subseteq \{0,1\}^n$ such that $\mathscr{X} = \mathscr{X}_0 \cup \mathscr{X}_1$ and $\mathcal{Y} = \mathcal{Y}_0 \cup \mathcal{Y}_1$, it holds that

$$\begin{aligned} L(\mathscr{X} \times \mathcal{Y}) &\leq L(\mathscr{X}_0 \times \mathcal{Y}) + L(\mathscr{X}_1 \times \mathcal{Y}) \\ L(\mathscr{X} \times \mathcal{Y}) &\leq L(\mathscr{X} \times \mathcal{Y}_0) + L(\mathscr{X} \times \mathcal{Y}_1). \end{aligned}$$

[8] The connection for formula complexity is implicit in [6], and is discussed explicitly in [8, 4, 2].

2.4 Subaddtive measures on protocol trees

We rely upon a property of subadditive measures defined on binary trees, like the protocol tree of a deterministic protocol Π. To this end, we define subadditive measures on binary trees as follows:

▶ **Definition 16.** Given a rooted binary tree $T = (V, E)$, we say that $\phi : V \to \mathbb{N}$ is a *subadditive measure on* T if for every vertex u with children v and w in T it holds that $\phi(u) \leq \phi(v) + \phi(w)$.

We state without proof the following easy lemma about such measures.

▶ **Lemma 17.** *Let $T = (V, E)$ be a rooted binary tree with root r and depth d, and let ϕ be a subadditive measure on T. Then there exists at least one leaf l of the tree for which, $\phi(l) \geq \left\lfloor \frac{\phi(r)}{2^d} \right\rfloor$*

2.5 Predictability and Average degree

In this paper we consider matrices of order $m \times n$ and partial matrices formed by a subset of rows, say σ, of the original set of rows, $[m]$. Let $S \subseteq (\{0,1\}^n)^\sigma$ denote a set of binary matrices whose rows are indexed by σ. We measure the information revealed on a typical row of S, conditioned on other rows, using the notion of predictability as in [1] but using the presentation in Raz-McKenzie [7].

▶ **Definition 18** (Projections)**.** Let $S \subseteq (\{0,1\}^n)^\rho$. Given a matrix $X \in S$ and a subset $\sigma \subseteq \rho$, we denote by X_σ the projection of X into rows indexed by σ. We extend the definition to a set of matrices S: $S_\sigma = \{X_\sigma \mid X \in S\}$

For a subset σ of $[m]$, we denote by $-\sigma$ its complement set in $[m]$, i.e., $[m] \setminus \sigma$. When σ is a singleton set say $\{i\}$, we denote it with i and $-\sigma$ with $-i$. Similarly, given a bit string $a \in \{0,1\}^m$ and a subset $\sigma \subseteq [m]$, we denote by $a_\sigma$the projection of a to coordinates in σ.

As mentioned earlier, we measure information using the layered graph corresponding to the set of matrices obtained as follows. We interpret a subset of matrices $S \subseteq \{0,1\}^{m \times n}$ as a $[m]$ bipartite graphs $G^i_S(U, V, E)$ for each $i \in [m]$, defined as follows. The left partition U is the set S_{-i}, the projection of S onto $[m] \setminus \{i\}$. The right partition V is the set S_i, the projection of S onto row i. The edge set E is defined as the set $\{(X, Y) \mid \exists Z \in S, X = Z_{-i}, Y = Z_i\}$. For any $X \in S_{-i}$, we denote by $\deg_i(X, S)$, the degree of the left node X in the graph G^i_S.

We use both average-degree and min-degree as information measures for measuring conditional information. More specifically, we use them to measure information on specific row, conditioned on the information about the other rows.

▶ **Definition 19** (Average degree of row)**.** Let S be a subset of $\{0,1\}^{m \times n}$. We define the average degree of the ith row in the set S to be,

$$\text{avgdeg}_i(S) = \frac{\sum_{X \in S_{-i}} \deg_i(X, S)}{|S_{-i}|} = \frac{|S|}{|S_{-i}|}.$$

Similarly we define min-degree as.

▶ **Definition 20** (Minimum degree of row)**.** Let S be a subset of $\{0,1\}^{m \times n}$. We define the average degree of the ith row in the set S to be,

$$\text{mindeg}_i(S) = \min_{X \in S_{-i}} \deg_i(X, S).$$

Collision probability denoted by $\mathrm{CP}_i(S)$ is defined to be the probability that two vectors chosen uniformly at random (with replacement) from S has the same ith element, i.e., $\mathrm{CP}_i(S) = \Pr_{X,X' \in S}[X_i = X'_i]$. The following is known about $\mathrm{CP}_i(S)$,

▶ **Lemma 21** (Lemma 4.4 from [1]). *$CP_i(S) \leq \frac{1}{\mathrm{avgdeg}_i(S)}$.*

In fact [1] proves something stronger. It also proves that for an individual $X \in S$, the collision probability with a randomly sampled X' is bounded by inverse of the average degree. We state it as the following lemma.

▶ **Lemma 22** (follows from proof of Lemma 4.4 of [1]). *For any $X \in S$, for an X' chosen uniformly at random (with replacement) from S has,*

$$\Pr_{X' \in S}[X_i = X'_i] \quad \leq \quad \frac{1}{\mathrm{avgdeg}_i(S)}.$$

The next lemma describes how average degree changes when we remove a set of matrices from S.

▶ **Lemma 23** (Lemma 4.5 from [1]). *Let $S' \subseteq S$,*

$$\mathrm{avgdeg}_i(S') \quad \geq \quad \frac{|S'|}{|S|}\mathrm{avgdeg}_i(S).$$

3 Depth lower bound

In this section we prove our main result. That is, we prove the following theorem, restated from the introduction.

▶ **Theorem 8.** *For any $m, n \in \mathbb{N}$, and any non-constant Boolean function $f : \{0,1\}^m \to \{0,1\}$ it holds that,*

$$CC(KW_{f \diamond U_n}) \quad \geq \quad \log L(f) + n - 2\log m - 10$$

where $L(f)$ is the formula complexity of the smallest formula computing f.

Let Π be a deterministic protocol solving the totalized variant of $KW_{f \diamond U_n}$ whose communication complexity is less than $\log L(f) + n - 2\log m + 10$ bits.

Recall that in the communication problem $KW_{f \diamond U_n}$, Alice gets (X, a) and Bob gets (Y, b) where X, Y are $m \times n$ matrices and a, b are m bit column vectors. We refer to X, Y as the matrix part, and a, b as the column vector part of players input, respectively. Recall that in the totalized variant of $KW_{f \diamond U_n}$, players can get inputs where the promises are not kept, and the protocol solving the totalized variant is supposed to reject such inputs.

We defer the techincal details of the argument to the appendix. The appendix section is organized as follows : In Appendix A we show the existence of a partial transcript on which the players have not learned too much information about their inputs. In Section B, we formalize the intuition that the information transmitted by an average partial transcript is divided between the matrix and the column vector part of the players input. In Section B.1, we do a cleanup to ensure that the players are forced to solve the second stage on unrevealed rows, i.e., rows in which the parties have learned very little information in the first stage.

References

1 J. Edmonds, R. Impagliazzo, S. Rudich, and J. Sgall. Communication complexity towards lower bounds on circuit depth. *computational complexity*, 10(3):210–246, Dec 2001. doi:10.1007/s00037-001-8195-x.

2 Dmitry Gavinsky, Or Meir, Omri Weinstein, and Avi Wigderson. Toward better formula lower bounds: an information complexity approach to the KRW composition conjecture. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 213–222, 2014.

3 Johan Håstad and Avi Wigderson. Composition of the universal relation. In *Advances In Computational Complexity Theory*, volume 13 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 119–134. DIMACS/AMS, 1990.

4 Mauricio Karchmer, Eyal Kushilevitz, and Noam Nisan. Fractional covers and communication complexity. *SIAM J. Discrete Math.*, 8(1):76–92, 1995.

5 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.

6 Mauricio Karchmer and Avi Wigderson. Monotone circuits for connectivity require super-logarithmic depth. *SIAM J. Discrete Math.*, 3(2):255–265, 1990.

7 Ran Raz and Pierre McKenzie. Separation of the monotone NC hierarchy. In *38th Annual Symposium on Foundations of Computer Science, FOCS '97, Miami Beach, Florida, USA, October 19-22, 1997*, pages 234–243, 1997.

8 Alexander A. Razborov. Applications of matrix methods to the theory of lower bounds in computational complexity. *Combinatorica*, 10(1):81–93, 1990.

9 G. Tardos and U. Zwick. The communication complexity of the universal relation. In *Proceedings of Computational Complexity. Twelfth Annual IEEE Conference*, pages 247–259, Jun 1997. doi:10.1109/CCC.1997.612320.

10 Andrew Chi-Chih Yao. Some complexity questions related to distributive computing (preliminary report). In *STOC*, pages 209–213, 1979.

A First Stage Communication

We let the players speak for $t_1 = \log(L(f)) - \alpha$ bits in the first stage, where α is a slack variable to be fixed later. We define a formal measure that measures the progress that the players make throughout the protocol towards solving $KW_{f \diamond U_n}$. We then show that this measure does not increase by much during the first stage.

We first define the set of matrices which are "alive" with respect to a partial transcript π of Π. We say that a matrix X is *alive* with respect to partial transcript π, if there exists a pair $(a,b) \in f^{-1}(0) \times f^{-1}(1)$ such that the input $((X,a),(X,b))$ is consistent with the partial transcript π. We define A_π^X as the set of all column vectors $a \in f^{-1}(0)$ for which (X,a) is an input for Alice consistent the partial transcript π. Similarly define B_π^X as the set of all column vectors $b \in f^{-1}(1)$ for which (X,b) is an input for Bob consistent with the partial transcript π. We let the set S_π be the set of alive matrices with respect partial transcript π. To quantify the total progress the players have made given a partial transcript π, we use the following measure.

$$\phi(\pi) \quad = \quad \sum_{X \in S_\pi} L\left(A_\pi^X \times B_\pi^X\right)$$

It is easy to note that the measure $\phi(\pi)$ defined above is a subadditive measure on the protocol tree of Π. By applying Lemma 17 on T' we get that, there exists a leaf $l \in T'$

and a corresponding partial transcript π' of Π, with $\phi(l) \geq \frac{1}{2^{\log(L(f))-\alpha}}\phi(\lambda)$. At the root $\phi(\lambda) = 2^{mn}L(f)$. Thus we get the following proposition,

▶ **Proposition 24.** *There exists a partial transcript π such that,*

- $\phi(\pi') \geq \frac{1}{2^{t_1}}2^{mn+\log(L(f))} = 2^{mn+\alpha}$.
- *Th length of π' is at most t_1, and length of π' is strictly less than t_1 if and only if π' is a leaf of the protocol Π.*

B Cleanup after first stage

In this section we do a cleanup so that players are forced to solve U_n on rows where not too much information was revealed.

At this point, we fix the partial transcript π' and differentiate between the information the players have learned about the matrices in $S_{\pi'}$, and the progress players have made on solving KW_f problem associated with an average $X \in S_{\pi'}$. To this end, we describe two measures of information. We first measure the information about the matrix. This is done by measuring what fraction of matrices from $\{0,1\}^{m\times n}$ are alive in $S_{\pi'}$. Throughout the argument we would like to discard some of the inputs which are consistent with the transcript, and measure the information with respect to remaining inputs. Hence, we also define these measures for arbitrary subsets of $\{0,1\}^{m\times n}$, and their associated rectangles.

Let $S \subseteq \{0,1\}^{m\times n}$ be such that every $X \in S$ has an associated rectangle $A_S^X \times B_S^X \subseteq f^{-1}(0) \times f^{-1}(1)$. The subadditive measure ϕ defined earlier extends naturally to such arbitrary as, $\phi(S) = \sum_{X\in S} L\left(A_S^X \times B_S^X\right)$. When $S = S_{\pi'}$, for every $X \in S$, $A_S^X \times B_S^X$ is defined to be $A_{\pi'}^X \times B_{\pi'}^X$. The information on the matrix part is measured as:

$$T_X^S \quad = \quad \log\left(\frac{2^{mn}}{|S|}\right) = mn - \log(|S|).$$

To measure the progress players have made on solving KW_f problem conditioned on an average X, we define:

$$T_{a|X}^S \quad = \quad \log\left(\frac{L(f)}{\frac{1}{|S|}\sum_{X\in S} L\left(A_S^X \times B_S^X\right)}\right) = \log(L(f)) - \log\left(\frac{1}{|S|}\phi(S)\right).$$

Observe that,

$$T_X^{\pi'} + T_{a|X}^{\pi'} \quad = \quad mn + \log(L(f)) - \log(\phi(\pi')). \tag{1}$$

Since we chose π' such that $\phi(\pi') \geq 2^{mn}2^{\alpha}$, we get that,

$$T_X^{\pi'} + T_{a|X}^{\pi'} \quad = \quad mn + \log(L(f)) - mn - \alpha = \log(L(f)) - \alpha.$$

Intuitively, this means that the total amount of information the players have learned in the first stage is bounded by the number of bits communicated in the first stage.

B.1 Regularization

In order to facilitate the analysis, we would like if the matrices $X \in S_{\pi'}$ has roughly the same formula complexity for its associated rectangle $A_\pi^X \times B_{\pi'}^X$ as an average X. In particular, we need that for every $X \in S_{\pi'}$, it holds that $L\left(A_{\pi'}^X \times B_{\pi'}^X\right) \approx 2^{\log(L(f))-T_{a|X}^{\pi'}}$. However, we do not have this property for every matrix X in $S_{\pi'}$. We thus construct a subset $S' \subseteq S_{\pi'}$, such that for every $X \in S'$, $L\left(A_{S'}^X \times B_{S'}^X\right) \approx 2^{\log(L(f))-T_{a|X}^{S'}}$ while maintaining that, $T_X^{S'} + T_{a|X}^{S'} \leq \log(L(f)) - \alpha + O(\log m)$.

We defer the details of the construction of S' to the full version. We denote the set S' by S^{reg}, signifying that this is a regularized set. From now on we work with the set S^{reg}.

B.2 Fixing revealed rows

In this section we describe an iterative process to classify the rows into revealed and unrevealed rows based on a threshold τ, and fix the corresponding bits in the column vector to be same for both players. In this process whenever we find a row whose average degree is smaller than $2^{n-\tau}$, we ensure that $a_i = b_i$ for this row. This specific row ceases to matter as we have maintained the promise irrespective of the value of this row for either players. We call such a row, identified by the iterative process and for which we fix $a_i = b_i$, an *inactive* row. Any row which is not an inactive row is called an *active* row. At any point during the iterative process, we would like to work with a set of matrices that consist only of the active rows. When we set $a_i = b_i$ for a new row i, we remove the row i from the set of active rows to the set of inactive rows. Now, formally, since the players expect to receive $m \times n$ matrices, we maintain a bijection between the set of matrices over active rows and the set of legal $m \times n$ matrices. Finally, when we remove a row from the set of active rows, we update the bijection. This is done by fixing the row as a function of the other rows. When the iterative process terminates, the set of active rows correspond to the unrevealed rows and the set of inactive rows correspond to the revealed rows with respect to threshold τ.

We fix a threshold τ on information learned about a row to classify the rows into "revealed" and "unrevealed rows". Under our average-degree measure, we say that a row is revealed with respect to a set of matrices $S \subseteq \{0,1\}^{m\times n}$, if $\text{avgdeg}_i(S) \leq 2^{n-\tau}$. Initially we look for revealed rows with respect to S^{reg}. We fix a revealed row to be a function of the remaining rows, and also fix the corresponding bit the in column vector. Fixing of a revealed row potentially reduces the set of alive matrices by eliminating the matrices whose extension into the revealed row is not according to the fixing chosen. The adversary proceeds by looking for revealed rows with respect to the current set and repeats the process of fixing such a row and corresponding column vector until there are no revealed rows with respect to the current set of alive matrices.

We have to make sure that the above process terminates before reducing the formula complexity of associated rectangles to zero. The following intuitive argument illustrates why the adversary can guarantee such a convergence. Since the players learned $T_X^{S^{\text{reg}}}$ amount of information about the entire matrix, when the adversary fixes a revealed row, the adversary accounts for τ amount of information from the total $T_X^{S^{\text{reg}}}$. However, fixing the corresponding bit in the column vector, could potentially reveal a bit of information about the remaining rows. Thus during one such operation, the adversary accounts for at least $\tau - 1$ bits of information from $T_X^{S^{\text{reg}}}$. Hence number of such operations, and consequently, the number of fixed rows can be at most $\frac{T_X^{S^{\text{reg}}}}{\tau-1}$. Since this is less than total number of rows, not all rows are fixed. We also need to ensure that at the end of all fixings every matrix has non-zero leaf complexity for its associated rectangle. We ensure that when fixing a bit corresponding to a revealed row, the formula complexity of the rectangle associated with a matrix reduces at most by a fraction of $\frac{1}{4}$th. Since every matrix X in S^{reg} had $2^{\log(L(f))-T_{a|X}^{S^{\text{reg}}}-1}$ formula complexity for the associated rectangle, we can ensure that after at at most $\frac{T_X^{S_i}}{\tau-1}$ fixings, the rectangle associated with a matrix has non-zero formula complexity.

The following lemma formalizes the effect of a single iteration of our iterative adversary.

▶ **Lemma 25.** *Let S' be a subset of $(\{0,1\}^n)^{m-|\sigma|}$, with an extension function $E_\sigma : S' \to \{0,1\}^{m\times n}$, along with rectangles $A_{S'}^X \times B_{S'}^X \subseteq f^{-1}(0) \times f^{-1}(1)$ for every matrix $X \in E_\sigma(S')$. Let $i \in [m] \setminus \sigma$ be a row such that $\text{avgdeg}_i(S') \leq 2^{n-\tau}$.*

Then there exists a set $S'' \subseteq S'_{-i}$, an extension function $E_{\sigma\cup\{i\}} : S'' \to \{0,1\}^{m\times n}$, along with rectangles $A_{S''}^X \times B_{S''}^X$ for every matrix $X \in E_{\sigma\cup\{i\}}(S'')$ such that,

- $\frac{|S''|}{2^{(m-(|\sigma|+1))n}} \geq \frac{|S'|}{2^{(m-|\sigma|)n}} 2^{\tau-1}$. *That is the amount of information in the matrix part, when restricted to rows* $[m] \setminus (\sigma \cup \{i\})$*, is decreased by* $\tau - 1$ *bits (from the amount of information in the matrix part, when restricted to rows* $[m] \setminus \sigma$*).*
- *For any* $X, Y \in E_{\sigma\cup\{i\}}(S'')$ *and for any* $a \in A^X_{S''}$ *and for any* $b \in B^Y_{S''}$ *it holds that* $a_i = b_i$*.*
- *For any* $X \in E_{\sigma\cup\{i\}}(S'')$*,* $L\left(A^X_{S''} \times B^X_{S''}\right) \geq \frac{1}{4} L\left(A^X_{S'} \times B^X_{S'}\right)$ *as long as* $L\left(A^X_{S'} \times B^X_{S'}\right) \geq 4$*.*

We defer the proof of the Lemma to the full version of the paper and prove Theorem 26 which shows how the adversary can iteratively fix revealed rows using Lemma 25 and get a set of matrices with the desired properties for the second stage.

▶ **Theorem 26.** *Let* $S^{reg} \subseteq \{0,1\}^{m\times n}$ *be the set from Section B.1. Then for any* $\tau \in \mathbb{N}$*,* $n > \tau > 3$*, there exists a set of indices* $\sigma \subseteq [m]$*, a subset* $S^{clean} \subseteq S^{reg}_{-\sigma}$*, an extension function* $E_\sigma : S^{clean} \to \{0,1\}^{m\times n}$ *and associated rectangles* $R^X_{S^{clean}} = A^X_{S^{clean}} \times B^X_{S^{clean}}$ *for every* $X \in E_\sigma\left(S^{clean}\right)$*, such that,*

- *The set* σ *of revealed rows is such that* $|\sigma| < m$*.*
- *For any row outside* σ *players have learned at most* τ *bits of information. More formally for any* $i \in [m] \setminus \sigma$*,*$\text{avgdeg}_i\left(S^{clean}\right) > 2^{n-\tau}$*.*
- *For any* $X \in E_\sigma\left(S^{clean}\right)$*,* $A^X_{S^{clean}} \times B^X_{S^{clean}}$ *is non-empty.*
- *For any* $X, Y \in E_\sigma\left(S^{clean}\right)$ *not necessarily distinct, for any* $a \in A^X_{S^{clean}}$*, and any* $b \in B^Y_{S^{clean}}$*, we have that* $a_\sigma = b_\sigma$*.*

Proof. The adversary uses Lemma 25 to fix revealed rows until there is none. Initially the adversary sets $\sigma \leftarrow \phi$, and $S' \leftarrow S^{\text{reg}}$. If there is any row $i \in [m] \setminus \sigma$, for which $\text{avgdeg}_i(S') \leq 2^{n-\tau}$, the adversary invokes Lemma 25 to obtain a set S'' of matrices, an extension function $E_{\sigma\cup\{i\}} : S'' \to \{0,1\}^{m\times n}$, and an updated set of fixed rows $\sigma \leftarrow \sigma \cup \{i\}$. Adversary sets $S' \leftarrow S''$, and checks again for a row $i \in [m]\setminus\sigma$, for which $\text{avgdeg}_i(S') \leq 2^{n-\tau}$. If no such row exists, the adversary stops. Otherwise the adversary invokes Lemma 25 on S' and repeats the procedure. Note that when the iterative process stops, the resulting set clearly satisfies the requirement.

We claim that the procedure terminates after at most $\frac{T^{S^{\text{reg}}}_X}{\tau-1}$ steps. Equivalently, $|\sigma| \leq \frac{T^{S^{\text{reg}}}_X}{\tau-1}$, as each invocation of the Lemma 25 increases $|\sigma|$ by 1. We use the following claim, whose proof is deferred to the full version, to show that $|\sigma| \leq \frac{T^{S^{\text{reg}}}_X}{\tau-1}$.

▶ **Claim 27.** *Between every invocation of Lemma 25, the following invariant holds,*

$$\frac{|S'|}{2^{mn-|\sigma|n}} \quad \geq \quad 2^{-\left(T^{S^{reg}}_X - |\sigma|(\tau-1)\right)}$$

Note that $S' \subseteq \{0,1\}^{(m-|\sigma|)\times n}$ and thus $\frac{|S'|}{2^{mn-|\sigma|n}}$ is always upper bounded by 1. This along with the invariant implies that, $1 \geq 2^{-\left(T^S_X - |\sigma|(\tau-1)\right)}$. That is $T^{S^{\text{reg}}}_X \geq |\sigma|(\tau-1)$. Since $\tau > 1$, this proves the claim that $|\sigma| \leq \frac{T^{S^{\text{reg}}}_X}{\tau-1}$. Since $T^{S^{\text{reg}}}_X + T^{S^{\text{reg}}}_{a|X} \leq \log(L(f)) - 4$, and since $\log(L(f)) \leq m$, we get that $|\sigma| < m$.

We claim that S' obtained at the end of such iterative fixing has all the properties claimed by the theorem for S^{clean}. That is we set $S^{\text{clean}} \leftarrow S'$.

Note that Lemma 25 ensures that the rows indexed by σ are fixed as a function of other rows for matrices in S^{clean}. The adversary stops invoking Lemma 25 on S' only when every row $i \in [m] \setminus \sigma$ has $\text{avgdeg}_i(S') > 2^{n-\tau}$. Hence every row $i \in [m] \setminus \sigma$ has $\text{avgdeg}_i\left(S^{\text{clean}}\right) > 2^{n-\tau}$.

Suppose at any point during the iterative fixing of the rows, every X that we consider, has formula complexity at least 4. Then, each invocation of Lemma 25 reduces the formula complexity of the associated rectangle by at most a quarter. Since we started with formula complexity $2^{\log(L(f))-T^{S^{\text{reg}}}_{a|X}-1}$ for every matrix X, after $|\sigma| \leq \frac{T^{S^{\text{reg}}}_X}{\tau-1}$ many invocations, any $X \in E_\sigma\left(S^{\text{clean}}\right)$, has formula complexity $2^{-\left(2\cdot\frac{T^{S^{\text{reg}}}_X}{\tau-1}\right)}$ fraction of the formula complexity in S^{reg}. That is,

$$\begin{aligned} L\left(A^X_{S^{\text{clean}}} \times B^X_{S^{\text{clean}}}\right) &\geq 2^{-\left(2\cdot\frac{T^{S^{\text{reg}}}_X}{\tau-1}\right)} L\left(A^X_{S^{\text{reg}}} \times B^X_{S^{\text{reg}}}\right) \\ &\geq 2^{\log(L(f))-T^{S^{\text{reg}}}_{a|X}-1-T^{S^{\text{reg}}}_X}. \end{aligned}$$

Note that this at least 2^3 as the statement of the theorem assumes that $T^{S^{\text{reg}}}_X + T^{S^{\text{reg}}}_{a|X} \leq \log\left(L\left(f\right)\right) - 4$ and $\tau > 3$. Thus in all of $|\sigma|$ invocations of the Lemma 25, all the rectangles associated with the all the matrices have formula complexity at least 8. This also ensures that for any $X \in E_\sigma\left(S^{\text{clean}}\right)$, $A^X_{S^{\text{clean}}} \times B^X_{S^{\text{clean}}}$ is non-empty at the end as $L\left(A^X_{S^{\text{clean}}} \times B^X_{S^{\text{clean}}}\right) \geq 8$. This concludes the proof the theorem. ◀

The fact that for any $X \in E_\sigma\left(S^{\text{clean}}\right)$,$L\left(A^X_{S^{\text{clean}}} \times B^X_{S^{\text{clean}}}\right) > 0$ implies that, there is at least one (a, b) pair in $A^X_{S^{\text{clean}}} \times B^X_{S^{\text{clean}}}$. From now on, for any $X \in E_\sigma\left(S^{\text{clean}}\right)$, we denote by $\left(a^X, b^X\right)$ an arbitrary but fixed (a, b) pair in $A^X_{S^{\text{clean}}} \times B^X_{S^{\text{clean}}}$.

C Second Stage Communication

In this section we describe how to handle the second stage of communication and complete the lower bound.

In the second stage, we let the players communicate at most $t_2 = n - \log m - 5$ bits of communication. At the end of the second stage the adversary would like to ensure that on any unrevealed row from the first stage, the players have not learned more than $\tau + 2 + t_2 = n - \log m$ bits of information. Intuitively this should be possible, because after the cleanup at the end of first stage, the players knew at most $\tau + 2$ bits of information about any unrevealed row in S''. To this end, we use Lemma 23 to ensure that with the additional t_2 bits of communication, the players learn at most t_2 bits of information on any row.

We consider the sub-tree $T''_{\pi'}$ of the protocol tree Π rooted at the node π' (the partial transcript from first stage). We would like to use Lemma 23 on the subtree $T''_{\pi'}$ to find a leaf of the protocol that is supported by at least 2^{-t_2} fraction of the matrices in S^{clean}. To this end, we need to prove that the depth of the tree is at most t_2. Recall that communication complexity of Π was less than $t_1 + t_2$, where t_1 is an upper bound on the number of bits communicated in the first stage. If π' itself is a leaf of the protocol Π, the tree $T''_{\pi'}$ is just a leaf and the claim is true. If π' is not a leaf of the protocol Π, the way we chose π' ensures that depth of π' is Π is equal to t_1. Hence there cannot be a leaf of $T''_{\pi'}$ which is at depth more than t_2, as the corresponding leaf in Π would be at depth more than $t_1 + t_2$, which is a contradiction to the fact $\text{CC}\left(\Pi\right) \leq t_1 + t_2$.

We define the following subadditive measure on the nodes of $T''_{\pi'}$. For a node $v \in T''_{\pi}$, corresponding to a partial transcript π'' of the second stage, define $\phi\left(\pi''\right) = |P_{\pi''}|$where $P_{\pi'} = S_{\pi''} \cap E_\sigma\left(S^{\text{clean}}\right)$. That is, $P_{\pi'}$ is the set of matrices consistent with π' and are from the set $E_\sigma\left(S^{\text{clean}}\right)$. It is easy to verify that this is indeed a subadditive measure on the protocol sub-tree $T''_{\pi'}$. Note that at the root of $T''_{\pi'}$, $\phi\left(\pi'\right) = \left|E_\sigma\left(S^{\text{clean}}\right)\right|$. Applying Lemma 17, we get that there is a leaf of T''_{π} corresponding to a transcript π'' of the second stage

for which $\phi(\pi'') \geq \frac{1}{2^{t_2}} \left|S^{\text{clean}}\right|$. Thus at the end of the second stage, we have a partial transcript π'' such that, $|P_{\pi''}| \geq 2^{-t_2} \left|S^{\text{clean}}\right|$. By Lemma 23, we get that for any $i \in [m]$, $\text{avgdeg}_i(P_{\pi'}) \geq 2^{-t_2}\text{avgdeg}_i\left(E_\sigma\left(S^{\text{clean}}\right)\right)$.

Hence at the end of the second stage, we have a full transcript and a set of inputs satisfying the following conditions.

▶ **Lemma 28.** *Let Π be any deterministic protocol solving $f \diamond U_n$. Let $t = t_1 + t_2$ where $t_1 = \log(L(f)) - \log m - 5$ and $t_2 = n - \log m - 5$. Then there are:*

- *A full transcript π of length at most t of Π.*
- *A subset $\sigma \subseteq [m]$ of indices where $|\sigma| < m$.*
- *A set $T \subseteq \{0,1\}^{(m-|\sigma|)\times n}$.*
- *An extension function $E_\sigma : T \to \{0,1\}^{m\times n}$.*
- *For any $X \in E_\sigma(T)$, a pair of column vectors a^X, b^X.*

satisfying the following,

- *For any $X, Y \in E_\sigma(T)$, $\left(X, a^X\right) \in S^A_\pi$ and $\left(Y, b^Y\right) \in S^B_\pi$.*
- *For any $X \in E_\sigma(T)$, $f\left(a^X\right) = 0$ and $f\left(b^X\right) = 1$.*
- *For any $X, Y \in E_\sigma(T)$, for any $i \in \sigma$, $a^X_i = b^Y_i$.*
- *For any row outside σ players have learned at most $t_2 + \tau$ bits of information. More formally for any $i \in [m] \setminus \sigma$, it holds that*$\text{avgdeg}_i(T) \geq 2^{n-\tau-2-t_2} = 2^{2+\log m}$.

C.1 Choosing the inputs

We show that the protocol Π must err on π, by showing that it must reject some inputs as well as accept some other inputs.

Since there is at least one input $((X,a),(X,b))$ consistent with π at the end of second stage, and since π is a leaf of the protocol Π, the transcript π has to reject.

We show that Π cannot reject on π by showing the existence of an input $\left(\left(X,a^X\right),\left(Y,b^Y\right)\right)$ consistent with π which satisfies all the promises. That is, for any $i \in [m] \setminus \sigma$, if $a^X_i \neq b^Y_i$ then $X_i \neq Y_i$. Recall that for $i \in \sigma$, we have already ensured that $a^X_i = b^Y_i$. We show that there exists inputs where for all $i \in [m] \setminus \sigma$, $X_i \neq Y_i$ using the probabilistic method. This would establish that, for any $i \in [m]$, if $a^X_i \neq b^Y_i$ then $X_i \neq Y_i$. By Lemma 21 we know that for two matrices U, V chosen uniformly at random (with replacement) from T, the probability that $U_i = V_i$ is at most $(\text{avgdeg}_i(T))^{-1}$. We set $X = E_\sigma(U)$ and $Y = E_\sigma(V)$. Thus for any $i \in [m] \setminus \sigma$, we have that $\Pr_{X,Y\sim E_\sigma(T)}[X_i = Y_i] \leq \frac{1}{2^{\log m+2}} = \frac{1}{4m}$. By a union bound, the probability for two matrices U, V chosen uniformly at random from T, the probability that all the rows indexed by $[m] \setminus \sigma$ are different is at least $1 - \frac{m-|\sigma|}{4m}$. For the rows indexed by σ, we already have that $a^X_i = b^Y_i$ for any $X, Y \in E_\sigma(T)$. Thus with probability $\frac{1}{4}$, the matrices X, Y satisfy all the promises when players are given the input $\left(\left(X,a^X\right),\left(Y,b^Y\right)\right)$. Hence the protocol cannot reject on π, establishing that Π errs on π.

Thus we get any protocol Π of length at most $t = \log(L(f)) + n - 2\log m - 10$ cannot correctly solve totalized variant of $KW_{f\diamond U_n}$, by choosing $\tau = 3$. This proves the main theorem from the introduction.

Round Complexity Versus Randomness Complexity in Interactive Proofs

Maya Leshkowitz
Weizmann Institute of Science, Rehovot, Israel
mleshkowitz@gmail.com

Abstract

Consider an interactive proof system for some set S that has randomness complexity $r(n)$ for instances of length n, and arbitrary round complexity. We show a public-coin interactive proof system for S of round complexity $O(r(n)/\log n)$. Furthermore, the randomness complexity is preserved up to a constant factor, and the resulting interactive proof system has perfect completeness.

2012 ACM Subject Classification Theory of computation → Interactive proof systems

Keywords and phrases Interactive Proofs

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.49

Related Version A full version of the paper is available at `https://eccc.weizmann.ac.il/report/2017/055/`.

Acknowledgements I would like to thank my advisor, Prof. Oded Goldreich, for his encouragement to approach the problem studied in this paper, and for his guidance and support in the research and the writing process. I also thank Dr. Guy Rothblum for useful discussions.

1 Introduction

The notion of interactive proof systems, put forward by Goldwasser, Micali, and Rackoff [8], and the demonstration of their power by Lund, Fortnow, Karloff, Nisan [12] and Shamir [14] are among the most celebrated achievements of complexity theory.

Loosely speaking, interactive proof systems capture the most general way in which one party can efficiently verify claims made by another, more powerful, party. The definition of interactive proof systems generalizes the traditional notion of a proof system (indeed, an NP-proof system), by allowing both *interaction* and *randomness*.

It is well known that both interaction and randomness are inherent to the power of interactive proof systems, where here we mean the extra power above that of NP-proof systems. Interactive proofs with no randomness can be easily transformed into NP-proof systems, whereas randomized non-interactive proofs, captured by the class $\mathcal{MA}$ (defined by Babai [1]), are essentially the randomized version of $\mathcal{NP}$ (e.g., loosely speaking, if $\mathcal{BPP} = \mathcal{P}$, then $\mathcal{MA} = \mathcal{NP}$).

Both *interaction* and *randomness* are quantitative notions; that is, one talks of the "amount of interaction", which is commonly associated with the (total) number of messages exchanged (a.k.a number of *rounds*), and of the amount of randomness (a.k.a *randomness complexity*). While the previous paragraph refers to the qualitative question and asserts that both interaction and randomness are essential, a finer study of the quantitative question is called for; that is, it is natural to ask about the necessary amount of interaction and randomness in various interactive proof systems.

The study of round-complexity in interactive proof systems has some well known results: Babai and Moran showed that the round complexity of any public-coin interactive proof system (a.k.a Arthur-Merlin proofs) can be reduced by a constant factor [2], whereas the public-coin transformation of Goldwasser and Sipser [9] (which essentially preserves the number of rounds) extends this result to general interactive proof systems. It is also known that a stronger round reduction is quite unlikely, since it would place SAT in co-AM-time($2^{o(n)}$), whereas AM-time(T) may equal Ntime(poly(T)). (This is the case due to a combination of results reviewed below in the paragraph titled "conditional tightness".)

In contrast, we are only aware of one study that focuses on the randomness complexity of interactive proof systems [1]. Specifically, Bellare et al. [3] studied the randomness complexity of *error reduction* in the context of interactive proof systems. (We mention that the randomness complexity is also of interest in [5], which provides an alternative transformation of general interactive proof systems to public-coin ones.)

Regarding the completeness in interactive proofs, the fact that every interactive proof system can be transformed into one with perfect completeness was shown in [4].

1.1 Round complexity versus randomness complexity

A natural question, which to the best of our knowledge was not considered before, is what is the relation between the two foregoing complexity measures. We do suspect that there are cases when the randomness complexity cannot be made smaller than the round complexity without trivially increasing the number of rounds. This is because constant-round interactive proof systems seem more powerful than NP-proof systems (see, e.g., the Graph Non-Isomorphism proof of [6]), whereas a logarithmic amount of randomness is clearly useless. But *can the randomness complexity be smaller than the round complexity?*

The answer is definitely negative if we consider *public-coin* interactive proof systems. Recall that in these proof systems, in each round, the verifier sends the outcome of fresh coins that it has tossed at the beginning of the current round, and so by definition the number of coin tosses is at least as large as the number of rounds.[2] However, it is not clear what happens in case of general interactive proofs. (In particular, the transformation of [9] significantly increases the randomness complexity by a factor depending on the number of rounds.)

Recall that in a general interactive proof system, the verifier may toss all coins at the very beginning, but its message in each round may be a complex function of the outcome of these coins (and the messages it has received from the prover). In particular, the verifier's messages may have very little information contents (from the prover's point of view), and so we may have many more rounds than the number of coins tossed. Furthermore, it is not clear how to collapse rounds that yield verifier messages of low information contents. These are the issues we deal with when showing that also in general interactive proof systems, randomness complexity $r(n)$ yields round complexity $O(r(n)/\log n)$, and hence in interactive proof systems *the round complexity is smaller than the randomness complexity.*

▶ **Theorem 1** (Main Theorem). *Suppose that S has an interactive proof system of randomness complexity $r(n)$ for instances of length n. Then, S has a* public-coin *interactive proof system*

[1] We refer to unconditional results and not to the long line of research of randomness versus hardness trade-off that rely on uniform or non-uniform assumptions, see, e.g. [10, 13].

[2] This presumes that the definition requires the verifier to send a non-empty message in each round. But otherwise (i.e., if the definition allows empty messages), rounds in which the verifier sends nothing can be collapsed.

of round complexity $O(r(n)/\log n)$ and randomness complexity $O(r(n))$. Furthermore, the resulting interactive proof system has perfect completeness.

Note that, in addition to obtaining the public-coin feature, we obtain perfect completeness for free. That is, even if the original system does not have perfect completeness, the new one has this feature.

We note that it is easier to prove a weaker version of the main theorem, which does not obtain the public-coin and perfect completeness features. The proof of this weaker result, outlined in Section 1.2, and given in detail in the full vesion of this paper [11, Apdx C], illustrates one of the ideas that underlie the proof of Theorem 1.

Conditional tightness

The round-complexity obtained by Theorem 1 is the best one may hope for at this time, since a result asserting round complexity $o(r(n)/\log n)$ for any set that has an interactive proof system of randomness complexity $r(n)$ would yield an unexpected result that conflicts with common beliefs and seems currently out of reach. Specifically, it would place $\mathtt{SAT}$ in co-AM-time$(2^{o(n)})$, which does contradict common beliefs. The full reasoning is as follows:

1. A variant of the celebrated interactive proof system for $\overline{\mathtt{SAT}}$ yields an interactive proof system of randomness complexity $O(n)$ for unsatisfiable CNFs with n variables. (This interactive proof consists of $n/\log n$ rounds such that in each round we strip a single variable in the sum-check that sums over $n/\log n$ variables with values in $[n]$, while using a finite field of size poly(n). See [7, Sec 8]).[3]
2. On the other hand, any set having an m-round interactive proof system is in AM-time$(n^{O(m)})$, see [7, Apdx B]. Hence, if unsatisfiable CNFs have an interactive proof of round complexity $o(n/\log n)$, then such instances can be refuted in AM-time$(2^{o(n)})$, whereas AM-time(T) may equal Ntime(poly(T)).

A different perspective

Theorem 1 may be viewed as an alternative transformation of general interactive proof systems into public-coin ones. Recall that the transformation of Goldwasser and Sipser [9] preserves the round-complexity of the original system (up to an additive constant), but increases the randomness complexity (i.e., raising it to a constant power). The same holds in the variant of that transformation presented in [5]. In contrast, Theorem 1 preserves the randomness complexity of the original system (up to a constant factor), but does not necessarily preserve the round-complexity. Taking this perspective, the fact that the round-complexity is bounded in terms of the randomness complexity is a consequence of the fact that the resulting scheme is of the public-coin type.

[3] This is done by packing a sequence of $\log n$ bits of the boolean variables into a symbol of $H = [n] \subseteq F$ where F is some field. For $i \in \ell$, where $\ell = \log n$, denote by $f_i(x) : F \to F$ the polynomial of degree $n-1$ that maps each element in H to the value of its ith boolean variable. Now, take the standard arithmetization of the CNF and replace each occurrence of the variable indexed $j \cdot \ell + i$ by the polynomial $f_i(x_j)$, where x_j is the F variable that represents the jth block of boolean variables. The resulting polynomial is a n/ℓ variable polynomial of total degree $O(m \cdot n)$, where m is the number of clauses. Finally, the number of satisfying assignments is given by the sum over all $(y_1, ..., y_{n/\ell}) \in H^{n/\ell}$ of the polynomial derived above. Furthermore, we do not execute the sum-check protocol over an exponentially large finite field but rather over a finite field of prime cardinality $p = \text{poly}(n)$, where p is selected by the verifier at random among such primes.

1.2 On the proof of Theorem 1

We start by giving an overview of a proof of a weaker result, in which we show how to transform any interactive proof, of randomness complexity $r(n)$, to a *private-coin* interactive proof for the same set that uses $O(r(n)/\log n)$ rounds, while maintaining the randomness complexity of $r(n)$. This proof gives a flavor of the proof of the main theorem, but is significantly simpler.

1.2.1 Private-coin emulation protocol

The idea of the emulation protocol is that, in every iteration, we would like the prover to send possible continuations of the current transcript (describing execution segments of possibly different number of rounds) that reveal much information about the verifier's random coins. Hence, the prover sends partial transcripts of *maximal* length such that each account for a large fraction of the residual probability mass[4]. The verifier then checks if one of these transcripts is consistent with the strategy determined by the values of its random coins, which were tossed upfront. If so, the verifier picks the maximal transcript consistent with its strategy and the verifier and prover proceed their interaction from that point. Otherwise, the verifier sends its next message (based on the aforementioned coins) without using the continuations suggested by the prover. We stress that the only source of the verifier's randomness is its private coins tossed upfront, which are used to determine the continuation of the transcript in each subsequent iteration.

We wish to elaborate on how the prover determines the continuations of the transcripts. Fixing an iteration, we denote the current transcript by γ and its residual probability mass by $p(\gamma)$. Each transcript the prover sends on this iteration is a possible continuation of γ of *maximal* length that is a "heavy continuation". By a heavy continuation γ', we mean that γ' has probability mass greater than $p(\gamma)/n$, when subtracting from it the probability mass of the continuations of γ that were either sent by the prover in previous iterations, or determined in this one.

This conditioning allows the prover to send several continuations of the transcript that are also continuations of each other. Consider for example the case that the prover sends $\gamma\alpha_1\beta_1\alpha_2\beta_2$ and $\gamma\alpha_1\beta_1$. In this case if the verifier chooses $\gamma\alpha_1\beta_1$ it means that in the next iteration the continuation of this transcript cannot begin with $\alpha_2\beta_2$.

The benefit of this method of determining transcript-continuations is that we guarantee that in the *next* iteration the probability mass of the new transcript is *lower* than $p(\gamma)/n$. The reasoning is as follows. If the verifier chooses one of the transcripts suggested by the prover, then on the *next* iteration the residual probability mass of each of its continuations is lower than $p(\gamma)/n$, otherwise this continuation should have been suggested by the prover on the previous iteration. If the verifier did not choose any of the transcripts, and instead continued the transcript with its own message $\widetilde{\alpha_1}$, then it follows that the residual probability mass of the transcript $\gamma\widetilde{\alpha_1}$ (under the conditioning of the appropriate events) is also lower than $p(\gamma)/n$, otherwise the continuation $\gamma\widetilde{\alpha_1}$ should have been suggested by the prover.

It follows that after $O(r(n)/\log n)$ rounds of interaction the probability of the transcript generated is at most $(1/n)^{r(n)/\log n} = 1/2^{r(n)}$, which means that there is a unique value of the coin tosses consistent with the transcript. Hence, a complete transcript is generated and

[4] Note that in the eyes of an observer, a verifier that samples its random coins at the beginning of the interaction and proceeds accordingly, is equivalent to a verifier that on each round samples a message with probability proportional to its residual probability mass.

the verifier can reject or accept at this point. It is easy to show that if the prover follows the prescribed emulation, then the verifier accepts with the same probability as in the original interactive proof system, and hence completeness is maintained.

Note that the above emulation per se does not suffice. It is essential to include validation checks that guarantee that the transcripts provided by the prover are consistent with some prover strategy for the original protocol. This means that if the prover provides two transcripts that share a prefix, this common prefix must end with a prover's message. This implies that the prover answers in the same way to the same verifier messages, which means that the prover's strategy is consistent with some prover of the original emulation, and so the soundness of the original proof system is maintained.

1.2.2 Public-coin emulation protocol

The simplified private-coin emulation protocol captures one of the key ideas of our public-coin emulation protocol. The difficulty that we face when seeking a public-coin emulation is that we cannot rely on hidden coins tossed upfront by the verifier. Thus, when presented with a list of heavy continuations, it is unclear how the verifier should select one at random, since the selection probability should be determined by the residual probability masses that are unknown to it. Our solution is to have the prover provide these probabilities, but this raises the need to verify these claimed values. (Needless to say, the verifier rejects upfront if the sum of these probabilities does not match the claimed probability of the transcript as determined before the current round.) In the last iteration, a complete transcript is sampled, containing the verifier's private coins, hence the validity of the transcript and the claim can be checked.

The foregoing description raises a few issues. Firstly, the prover should find a way to communicate all the transcripts to the verifier, and not only the ones with high residual probability mass as before. Second, it is not clear what happens when the prover provides wrong values for the residual probabilities. As for the second issue, note that maliciously raising the probability of a transcript does contribute towards having the sum of probabilities meet the prior claim, but it makes the probability that this transcript is selected higher, and so puts the prover in greater problem in the next round. Indeed, a careful analysis shows that actually the prover gains nothing by such behaviour, since when the transcript is complete, false claims about its residual probability are easily detected.

Turning back to the first issue, we note that the issue is that there may be too many short transcripts that each account for a small fraction of the residual probability mass. To deal with this case, we pack many transcripts into a single auxiliary message, which means that we use a succinct representation of a sequence that contains many of the transcripts but not all of them (since otherwise we would have made no progress at the current round). The succinct representation should support the verification that the corresponding sequences are disjoint. Now, each such "pack" of transcripts will be assigned the corresponding probability mass, and be treated as if it were an actual transcript.

Needless to say, the foregoing is but a very rough sketch of the structure of the derived proof system. The actual proof system uses a carefully designed verification procedure that ensures that its executions can be mapped to executions in the original proof system.

We note that while the above description of the public-coin emulation refers to the probability that various transcripts appear in the original proof system (when the prover uses an optimal strategy), our actual construction refers only to accepting transcripts (i.e., transcripts that lead the original verifier to accept). Consequently, we obtain a proof system of perfect completeness, even if the original proof system had two-sided error probability.

1.3 Organization

Towards proving the main theorem we shall show how to emulate an existing interactive proof system with a public coin emulation protocol that has $O(r(n)/\log n)$ rounds. We begin by introducing the notion of "protocol trees" in Section 3, which we use to describe the interaction of the verifier and prover of the original interactive proof system. In Section 4, we shall show how to transform the protocol tree into an "emulation tree", that contains the continuations of the transcripts that the prover sends on each iteration along with their probability masses. Using this emulation tree, we then turn to describing the public-coin emulation protocol for the new prover and verifier in Section 5. The analysis of the emulation, which is partitioned into completeness and soundness, is given in the full version of the paper [11, Sec 6]. Also given in the full version [11, Apdx C] is a private-coin emulation protocol for emulating the interactive proof system and its analysis, which is less involved than the public-coin emulation.

2 Preliminaries

Let us start by formally defining interactive proof systems, where the completeness and soundness bounds are parameters.

▶ **Definition 2** (Interactive Proof Systems)**.** Let $c, s : \mathbb{N} \to [0,1]$ such that $c(|x|) \geq s(|x|) + \frac{1}{poly(|x|)}$. An interactive proof system for a set S is a two party game, between a verifier executing a probabilistic polynomial time strategy, denoted V, and a prover executing a *(computationally unbounded)* strategy, denoted P, satisfying the following two conditions:

- Completeness with bound c: For every $x \in S$, the verifier V accepts after interacting with the prover P on common input x with probability at least $c(|x|)$.
- Soundness with bound s: For every $x \notin S$ and every prover strategy $\widetilde{P}$, the verifier V accepts after interacting with $\widetilde{P}$ on common input x with probability at most $s(|x|)$.

When c and s are not specified, we mean $c \equiv 2/3$ and $s \equiv 1/3$. We denote by $\mathcal{IP}$ the class of sets having interactive proof systems. When $c \equiv 1$, we say that the system has perfect completeness.

3 The protocol tree of the original proof system

Fixing an interactive proof and an instance x of length n, we describe the possible prover-verifier interactions of the system on common input x using a tree whose height corresponds to the number of rounds of interaction. For some $\ell = \ell(n)$, we assume without loss of generality that in each round the verifier sends a message $\alpha \in \{0,1\}^\ell$, and the prover's responds with a message $\beta \in \{0,1\}^\ell$. We can also assume, without loss of generality, that the prover's strategy is deterministic and fixed. Each node v in level j represents a possible prover-verifier transcript for the first j rounds of the interaction. The branching of the tree represents the possible ways to extend the transcript to the next round. The number of ways to extend the transcript depends only on the verifier's message, since we fixed the prover's strategy. Hence, each node has *at most* $d \coloneqq 2^\ell$ children, corresponding to the 2^ℓ possible verifier messages for the next round. The prover's response to each such message is included in the **description** of the corresponding node.

The description of a node u on level j contains the partial **transcript** $\gamma(u) = \alpha_1\beta_1, \ldots, \alpha_j\beta_j$ of the interaction up to the j'th round. The root (at level zero) has an empty transcript, whereas a leaf of the tree represents a complete prover-verifier interaction.

We can assume, without loss of generality, that the verifier sends its private coins on the last round, and hence every leaf is associated with a sequence of coin tosses which either leads the verifier to accept or to reject. Hence, we can represent the possible interactions generated by the interactive proof system using a tree of height m which has $2^{r(n)}$ leaves, where m is the number of rounds and $r(n)$ is the number of coin tosses. Using a constant number of parallel repetitions, we can assume that the interactive proof system has completeness parameter $\frac{9}{10}$ and soundness parameter $\frac{1}{10}$. Note that this blows up the randomness complexity only by a constant factor (as compared to our interactive proof for the standard $\frac{1}{3}$, $\frac{2}{3}$ parameters). Therefore, if x is a yes-instance then at least $\frac{9}{10} \cdot 2^{r(n)}$ of its leaves represent accepting runs, and if x is a no-instance then at most $\frac{1}{10} \cdot 2^{r(n)}$ of its leaves represent accepting runs.

The description of a node also contains its **weight**, denoted $w(u)$. The weight of the node is the number of coin sequences that are consistent with the node and lead the verifier to accept at the end of the interaction. That is,

▶ **Definition 3** (Weight of a leaf)**.** Let u be a leaf with transcript $\gamma(u)$ which corresponds to the full transcript of the interaction of P and V on input x, when V uses coins ρ; that is,

$$\gamma(u) = (\alpha_1, \beta_1, \ldots, \alpha_m, \beta_m, (\rho, \sigma)) \tag{1}$$

where $\sigma = V(x, \rho, \beta_1, \ldots, \beta_m) \in \{0, 1\}$ is V's final verdict and for every $i = 1, \ldots, m$ it holds that $\alpha_i = V(x, \rho, \beta_1, \ldots, \beta_{i-1})$ and $\beta_i = P(x, \alpha_1, \ldots, \alpha_i)$. We define the weight $w(u)$ of u to be V's final verdict σ.

▶ **Definition 4** (Weight of a node)**.** The weight $w(u)$ of a node u in the protocol tree is the sum of the weights of the leaves that are descendants of u.

Note that $w(u)$ is proportional the probability that $\gamma(u)$ is generated and the verifier accepts at the end of the interaction.

4 The emulation tree

4.1 Overview

So far we explained how to represent the possible executions of a m-round interactive proof system on some instance x, where the protocol utilizes $r(n)$ coins. This resulted in a protocol tree of height m with $2^{r(n)}$ leaves. Our goal is to transform this protocol tree to an emulation tree that defines a prover strategy for a $O(r(n)/\log n)$-round public-coin emulation protocol. This transformation is done using the Build_Tree procedure. First we describe a very restricted case where the protocol tree is already suitable for our proposed public-coin emulation and the Build_Tree procedure is not required. Next, we explain how the transformation works in a restricted case when the degree of the protocol tree is bounded by poly(n), and finally in the case of a general protocol tree.

The protocol tree is of height $O(r(n)/\log n)$ and degree poly(n)

In order to convince the verifier that x is a yes-instance the prover makes an initial claim that the weight of the root of the protocol tree is at least $c \cdot 2^{r(n)}$ (where c is the completeness bound). The emulation is initiated at the root of the protocol tree and on each round of the emulation the prover assists the verifier at progressing one step down the protocol tree. (This assistance is required because the verifier does not have access to the protocol tree.) Each round consists of the prover providing the verifier with the descriptions of the children of the current node u that was sampled on the previous round, where these descriptions

contain the weights of the various children. The verifier performs validations to check that according to the descriptions these are legal children of u, and that their claimed weights sum up to $w(u)$. Then, the verifier samples a child with probability that is approximately proportional to its weight, up to a multiplicative factor of $1 + \frac{1}{n}$, using $O(\log n)$ public coins. On the last round, a leaf is sampled, whose description contains the complete prover-verifier interaction along with the coins tossed by the verifier. The new verifier accepts if and only if the transcript sampled is consistent with the original verifier's strategy according to the coin tosses revealed, and leads the original verifier to accept.

To see why this is indeed an interactive proof system for the original language, note that an honest prover can always convince the verifier of the correctness of a true claim using this emulation. Hence the interactive proof system we described has perfect completeness. On the other hand, for no-instances, a prover that wants to make the verifier accept must make an initial claim that the weight of the empty transcript is much larger than its real weight. Namely, the real weight of the empty transcript is at most $s \cdot 2^{r(n)}$, whereas the prover claims that the weight is at least $c \cdot 2^{r(n)}$, where $c > s$ are the completeness and soundness bounds of the original interactive proof system. Thus, there is a multiplicative gap of $\frac{s}{c}$ between the real weight and the one claimed. We can show that, in expectation, this gap is maintained throughout the emulation, up to a factor of $(1 + \frac{1}{n})^n$ that comes from the approximation factor. Therefore, the probability that a leaf that corresponds to an accepting run is sampled on the last round (and hence the verifier accepts) is at most $\frac{s}{c}(1 + \frac{1}{n})^n$, which is smaller than $\frac{1}{3}$ for a suitable choice of s and c.

The degree of the protocol tree is bounded by poly(n)

In this case the height of the protocol tree may be asymptotically larger than $\frac{r(n)}{\log n}$. We create a new tree of height $O(r(n)/\log n)$ to guide the prover's strategy, which we use in a way similar to how we used the protocol tree in the previous paragraph. We call this tree the **emulation tree**. The nodes in the emulation tree are nodes from the protocol tree, however the children of a node u in the emulation tree may be non-immediate descendants of u in the protocol tree.

We start with a protocol tree T rooted at r whose weight is $w(r)$, and on each step we modify this tree towards creating an emulation tree. We define a **heavy descendant** of r to be a node in T whose weight is at least $\frac{w(r)}{n}$, and the weight of each of its children is smaller than $\frac{w(r)}{n}$. Note that there are at most n such nodes.

We modify T so that the children of r in the emulation tree are its original children as well as the heavy descendants that we lift upwards to make them new children of r. This modification is performed using the Build_Tree(r) procedure, which when invoked on a node r identifies the nodes that will be children of r in the new tree, sets them as children of r, and then initiates recursive invocations on the (original and new) children of r, creating the new emulation tree rooted at r. Details follow.

Let T_r denote the temporary tree after the stage that we identify and set the children of r. We start by identifying the heavy descendants of r (they can also be children of r in T), which will become **heavy children** of r in T_r. We then proceed to the non-heavy children of r in T, which will also be children of r in T_r. After we identified the children of r in the new emulation tree, we call the Build_Tree procedure on each child of r in T_r, which creates an emulation tree rooted at that node.

Observe that in the final emulation tree, for each node u, the weight of each grandchild of u is at most $\frac{w(u)}{n}$. This is because if v is an heavy child of u in the emulation tree, then the weight of the descendants of v in T_u is at most $\frac{w(u)}{n}$. Since the children of v in the emulation tree are descendants of v in T_u the claim holds. Otherwise, v is non-heavy child of u, so its

weight is smaller than $\frac{w(u)}{n}$ and hence the weight of the children of v is also smaller than $\frac{w(u)}{n}$.

It follows that the height of the final emulation tree is $O(r(n)/\log n)$. The number of children of each node is at most $\text{poly}(n)$, because we add at most n heavy children to the original children of each node. Hence the emulation tree has properties similar to the protocol tree in the previous paragraph, so it is suitable for our public-coin emulation described in the previous subsection.

The general case

In general, the degree of the protocol tree is unbounded, and hence it may be exponential in n. Lifting the heavy children as we did in the case of unbounded height guarantees that the height of the new emulation tree is $O(r(n)/\log n)$, but its degree may be super-polynomial in n (due to the original children). Hence, we will not be able to perform the emulation. Thus, we also need to make sure the degree of the new tree is $\text{poly}(n)$.

In order to reduce the degree when it is too large, we group the non-heavy children of r under new children, which we call **interval children**. This is done in addition to handling the heavy children of r as before. In general, children of r in T may become non-immediate descendants in T_r, and non-immediate descendants of r may become immediate children of r in T_r (due to lifting).

Determining the children of r in T_r is done in two steps, as part of the Build_Tree(r) procedure. The first step is identifying the heavy descendants of r and lifting them to be children of r, creating heavy children in T_r. Next, we unite the non-heavy children of r into groups. We unite the children by lexicographic order of their transcript field, such that the weight of each group is larger than $\frac{w(r)}{n}$ and at most $\frac{2w(r)}{n}$ (except for, possibly, the last group which is only required to have weight smaller than $\frac{2w(r)}{n}$). We create a new **interval child** v for each such group, where the children of v are the nodes in the group.

After this step, the children of r in the final emulation tree are exactly the children of r in T_r. The number of children r has is at most n, since the weight of each child of r (except for possibly the last interval child) is at least $\frac{w(r)}{n}$. Next, the procedure is called recursively on the children of r in T_r in order to create the final emulation tree.

The description of a node u in the emulation tree is composed of the **transcript** field $\gamma(u) = \alpha_1\beta_1 \ldots \alpha_i\beta_i$ and a **weight** field $w(u)$ as in the original protocol tree, with an additional **range** field $R(u)$. The range of a node represents the possible range of its children's transcripts. After determining the heavy children of u, and before grouping the non-heavy children under interval children, the non-heavy children of u are all children of u in the original tree. Hence, the transcripts of the children of each interval child are the same up to the last verifier's message on which they differ, which corresponds to the branching of the protocol tree for the next round. Thus, we can label the range $R(u) = [s, e]$ where $s < e \in \{0,1\}^\ell$ according to the range of the last verifier message in the transcript field of the children of u. Heavy children have full range $\left[0^\ell, 1^\ell\right]$, whereas the range of interval children is a subinterval of $\left[0^\ell, 1^\ell\right]$ such that this subinterval corresponds to the transcripts of the descendants that are grouped under this node.

We show in the analysis that the height of the final emulation tree is $O(r(n)/\log n)$. The degree of nodes in the final emulation tree is at most n, hence it is suitable for public-coin emulation like in the previous subsection.

(The running time of this algorithm is at least the size of the protocol tree, which is exponential in $r(n)$, and thus it may be exponential in $|x|$. However, the prover is the one that runs this algorithm and the prover is computationally unbounded. Therefore the running time is not an issue.)

4.2 The Build Tree procedure

Denote the designated prover and verifier of the original interactive proof system by P_0 and V_0 respectively, and the protocol tree of P_0 and V_0 for a yes-instance x by T_{P_0,V_0}. The Build_Tree procedure is a recursive procedure that reads and updates a global tree T, which is initially set to equal the protocol tree T_{P_0,V_0} until obtaining the final emulation tree, denoted by E_{P_0,V_0}. When invoked on a node u in T, the procedure determines the children of u, updates the global tree and invokes the procedure recursively on the children of u.
We denote by $T(u)$ the subtree of T rooted at u.

Initialization

The tree T is initialized to be the original protocol tree, where each node has a description that contains the weight and transcript like in the original tree, and an additional range field which is initially left empty. We set the range of the root, denoted r, to be full range $R(r) = [0^\ell, 1^\ell]$. If the weight of r is zero we terminate the process. Otherwise, we invoke the Build_Tree procedure on r.

The main procedure: Build_Tree

If u is a leaf, the procedure returns without updating the global tree T. Otherwise, the Build_Tree procedure invokes two sub-procedures, Build_Heavy(u) and Build_Interval(u), in order to identify and update the children of u in T. Finally, the Build_Tree procedure is invoked recursively on all the children of u in T.

Build_Heavy

The Build_Heavy procedure identifies the **heavy descendants** of u in $T(u)$, which are descendants of large weight that have no children of large weight, and modifies the tree by lifting them to become **heavy children** of u.

▶ **Definition 5** (Heavy descendants)**.** We call v a heavy descendant of u if v is a descendant of u in T and the following conditions hold:

1. $w(v) \geq \frac{w(u)}{n}$
2. Either v is a leaf, or for each child z of v it holds that $w(z) < \frac{w(u)}{n}$.

For each heavy descendant, v, of u we perform the following process:

1. Update v's description: Set the range field of v to be full range $R(v) = [0^\ell, 1^\ell]$.
2. Modify the protocol tree if v is not already a child of u:
   a. Subtract $w(v)$ from the weight of the ancestors of v in $T(u)$, except for u whose weight stays the same.
   b. Move v (along with the subtree rooted at v) to be directly under u.

See Figure 1.
After we finish identifying and moving the heavy children of u we perform a clean up stage where we erase all the nodes in T with weight zero.

Build_Interval(u)

This procedure groups the non-heavy children of u under **interval children**. Denote the range field of u by $R(u) = [s(u), e(u)]$. (Note that $s(u) = 0^\ell$ and $e(u) = 1^\ell$ unless u is an interval node, in which case its range is partial.) We partition the range of u into a sequence

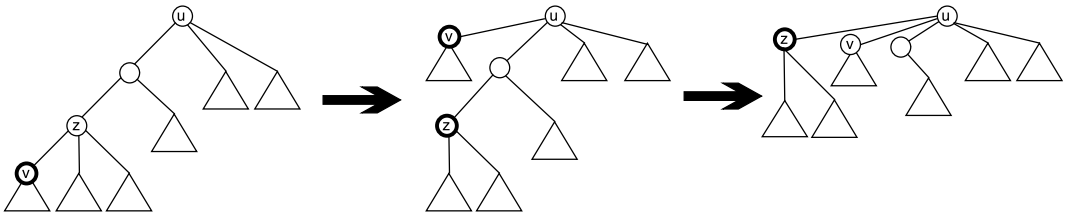


Figure 1 Build_Heavy: In the first step, v is identified as a heavy descendant of u and moved to be a heavy child of u. In the second step, z is identified and moved to be a heavy child of u. The triangles represent subtrees of the original tree.

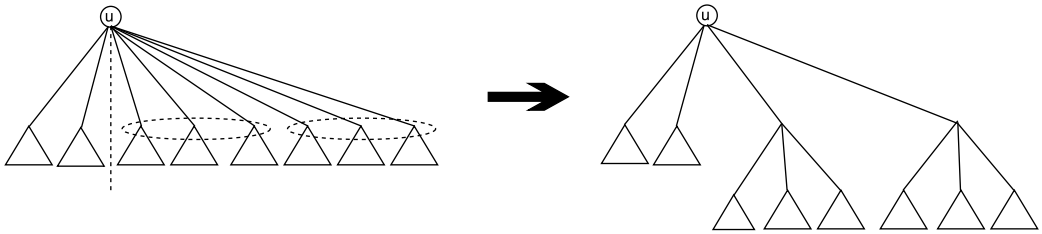


Figure 2 Build_Interval: The left diagram represents the tree before the Build_Interval procedure. The nodes to the left of the dashed line are heavy children of u. The group of nodes inside each dashed circle are united under an interval node. The tree on the right is the result of applying the Build_Interval procedure.

of consecutive intervals, each one representing the range of a new child of u. As long as we have not partitioned all of the range $[s(u), e(u)]$ we perform the following procedure.

1. Determine s', the starting point of the interval child's range: Initially, for the first interval child of u we set $s' = s(u)$. For the next interval children, if the end of the range of the previously created interval child is $\tilde{e}$, then we set $s' = \tilde{e} + 1$.
2. Determine e', the ending point of the interval child's range: For each $e \in \{0,1\}^\ell$, denote by $non\_heavy(s', e)$ the set of children of u in $T(u)$ whose weights are smaller than $\frac{w(u)}{n}$ and their last verifier message α (in the transcript field) is in the range $[s', e]$. Note that when $[s', e] \neq [s(u), e(u)]$ the set $non\_heavy(s', e)$ can be a proper subset of set of non-heavy children of u. We define the weight of the set $non\_heavy(s', e)$, which we denote by $W(s', e)$, as the sum of the weights of nodes in $non\_heavy(s', e)$.
   We set e', to be the minimal $e \in \{0,1\}^\ell$ that satisfies $W(s', e) \geq \frac{w(u)}{n}$. If no such e exists and $W(s', e(u)) > 0$, we set $e' = e(u)$. If $W(s', e(u)) = 0$ there is no need to create another interval child so we return to the Build_Tree procedure. (This guarantees that the weight of an interval child is at least 1).
3. Create a new node v: We set the transcript of v to be like the transcript of u, $\gamma(v) = \gamma(u)$, its range to be $R(v) = [s', e']$ and its weight to be $w(v) = W(s', e')$.
4. Place v in the tree: disconnect u from the nodes in $non\_heavy(s', e')$. Set u as a parent of v and let v be the parent of all nodes that are in $non\_heavy(s', e')$.

See Figure 2. Note that the weight of an interval child of u is at most $\frac{2w(u)}{n}$ and at least $\frac{w(u)}{n}$, except possibly for the last interval child, whose weight is at least 1.

4.3 Properties of the emulation tree

Recall that in the original protocol tree, v was a child of u if and only if $\gamma(v) = \gamma(u)\alpha\beta$ where $\alpha, \beta \in \{0,1\}^\ell$ denote the next verifier message and the prover's response to it. However, in the new emulation tree, E_{P_0,V_0}, this is not the case. Namely, if v is a child of u in E_{P_0,V_0},

then it could be that v is an heavy child of u and hence $\gamma(v) = \gamma(u)\alpha_1\beta_1, \ldots, \alpha_k\beta_k$ for some $\alpha_1, \beta_1, \ldots, \alpha_k, \beta_k \in \{0,1\}^\ell$, or v is an interval child hence $\gamma(v) = \gamma(u)$ and $R(v) \subseteq R(u)$. Nevertheless, the following properties of the final emulation tree E_{P_0,V_0} hold. The proofs can be found in the full version of the paper [11].

▶ **Claim 6** (Node degree). *Each node u in the final emulation tree E_{P_0,V_0} has at most n children.*

▶ **Claim 7** (Weight reduction). *For every node u in the final emulation tree E_{P_0,V_0} the weight of each grandchild of u in E_{P_0,V_0} is at most $\frac{2w(u)}{n}$.*

▶ **Corollary 8** (Corollary to Claim 7). *The height of the final emulation tree E_{P_0,V_0} is $O(r(n)/\log n)$.*

▶ **Claim 9** (Leaves). *The leaves of E_{P_0,V_0} are exactly the leaves of protocol tree T_{P_0,V_0} whose weights are 1.*

5 Public-coin emulation

Next, we describe the strategy of the designated prover P and verifier V in the new ("emulation") protocol. The strategy of the designated prover P for a yes-instance x uses the emulation tree E_{P_0,V_0} of x constructed in the previous section. The prover assists the verifier V in progressing down the emulation tree. On each iteration, the prover provides the descriptions of the children $v_1, \ldots, v_d$ of the current node u, which was sampled in the previous iteration. The verifier preforms validations on the list supplied by the prover (to be detailed below), and then samples one of the children for the next iteration according to its weight. The verifier does not have access to the emulation tree, and its validations consist of structural requirements on the part of the emulation tree seen so far. On the last round the verifier checks that the full transcript, along with the sequence of coin tosses, leads the original verifier V_0 to accept.

One of the structural validations that the verifier makes is that the nodes provided by the prover may be children of u in the emulation tree. For nodes in the original protocol tree, v is a child of u if and only if the transcript of v extends the transcript of u by one pair of messages, and thus v is a descendant of u if and only if the transcript of u is a proper prefix of the transcript of v. For nodes in the emulation tree the situation is more complex. If v is an interval child of u, then the transcript of v equals the transcript of u, and the range of v is a partial range of the range of u. If v is a heavy child of u, then the transcript of u is a proper prefix of the transcript of v. Furthermore, if $\gamma(u) = \alpha_1^u\beta_1^u, \ldots, \alpha_i^u\beta_i^u$, then α_{i+1}^v (the $i+1$ verifier's message in the transcript of v) should be in the range of u.

With these two cases in mind, we define the conditions required of the descriptions of two nodes u and v in order for v to be a descendant (not necessarily a child) of u in the emulation tree. We say that v is a **transcript descendant** of u if these required conditions hold.

▶ **Definition 10** (Transcript Descendant). Denote by u and v nodes in the emulation tree with transcripts $\gamma(u) = \alpha_1^u\beta_1^u, \ldots, \alpha_i^u\beta_i^u$ and $\gamma(v) = \alpha_1^v\beta_1^v \ldots \alpha_j^v\beta_j^v$ and with range field $R(u)$ and $R(v)$, respectively. We say that v is a transcript descendant of u if one of the following conditions hold:

(i) $\gamma(v) = \gamma(u)$ and $R(v) \subseteq R(u)$

(ii) $\gamma(u)$ is a proper prefix of $\gamma(v)$ and $\alpha_{i+1}^v \in R(u)$

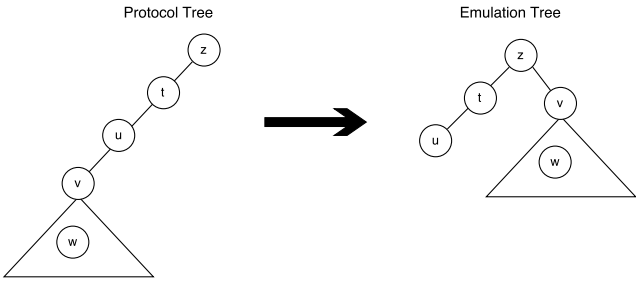


Figure 3 Truncation during Build_Tree. The node v is identified as a heavy descendant of z, so it is moved (along with the subtree that is rooted at it) to be a heavy child of z.

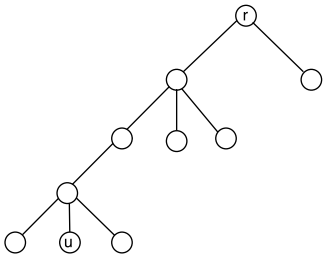


Figure 4 The nodes in the list of seen nodes, S, in the iteration that the input node is u.

Note that it is possible that v is a descendant of u in the emulation tree and the description of u is equal to the description of v. For example, this can happen when v is the only interval child of u.

It is easy to show that for every node u in E_{P_0,V_0}, every descendant of u is a transcript descendant of u. The proof is similar to the proof in the completeness part of the analysis [11, Sec 6].

We state the following claim regarding the transitivity property of transcript descendancy, which we use in the analysis of the emulation.

▶ **Claim 11** (Transitivity). *For nodes u, v and z in the emulation tree such that z is a transcript descendant of v and v is a transcript descendant of u, z is a transcript descendant of u.*

The proof of the claim is given in [11, Apdx A]. It follows by case analysis of the different transcript descendancy types between u, v and z.

The condition of v being a transcript descendant of u is not sufficient to guarantee that v may be a descendant of u in the emulation tree E_{P_0,V_0}. For example, suppose that v was a child of u in T_{P_0,V_0}, and is a heavy descendant of a node z that is an ancestor of u in E_{P_0,V_0}. Then, v becomes a heavy child of z in E_{P_0,V_0}, which means that the subtree rooted at v was truncated and moved up to be a direct descendant of z. Therefore, in order to check if v may be a legal descendant of u in the emulation tree, the verifier needs to check that v does not belong to a part of the tree that was truncated and moved to a different part of the emulation tree (such as nodes v and w in Figure 3). For this reason the verifier keeps a list S of nodes that were seen during the emulation, and updates the list at every iteration with the new nodes seen. Note that the nodes in S, which the verifier sees up to some iteration, are a subtree of the emulation tree that is composed of a path from the root of the tree to the current input node, augmented with the children of the nodes in the path. See Figure 4.

Another structural validation requires the transcripts that the verifier sees during the execution to be consistent with a deterministic prover strategy.

▶ **Definition 12** (Prover consistent)**.** We say that two transcripts $\gamma(u)$ and $\gamma(v)$ are **prover consistent** if the maximal prefix they agree on is either empty or ends with a prover's message. That is, the prover should respond in the same way on the same prefix of the transcripts (i.e., for every j smaller than the length of the shorter transcript, if $(\alpha_1^u\beta_1^u,\ldots,\alpha_j^u)=(\alpha_1^v\beta_1^v,\ldots,\alpha_j^v)$ then $\beta_j^u=\beta_j^v$).

This condition will allow for extracting a prover's strategy for the original protocol from the transcripts in E_{P_0,V_0}, and then to claim that since the original prover cannot fool the verifier with high probability, the new prover cannot either.

Initially, for the first iteration, the transcript of the root r is the empty transcript and the range is full range, $[0^\ell,1^\ell]$. The prover provides the weight of the root r and the verifier checks that the claimed weight is at least $\frac{9}{10}\cdot 2^{r(n)}$. The verifier adds the description of r to the list of seen nodes S. The rest of the first iteration, as well as subsequent iterations, proceed as follows.

▶ **Construction 13** (the ith iteration)**.** On input a non-leaf node u and list S.[5]

1. The prover provides the descriptions of the children $v_1,\ldots,v_d$ of u:

$$(\gamma(v_1),R(v_1),w'(v_1)),\ldots,(\gamma(v_d),R(v_d),w'(v_d))$$

2. The verifier preforms the following validations and rejects if any of them fails:

a. The verifier checks that all nodes are different (according to their descriptions) that is, for each distinct $i,j\in[d]$, if $\gamma(v_i)=\gamma(v_j)$, then $R(v_i)\neq R(v_j)$.

b. The verifier checks that the weights of the children of u sum up to $w'(u)$; that is,

$$w'(u)=\sum_{j=1}^{d}w'(v_j) \tag{2}$$

c. For each $j\in[d]$, the verifier checks that v_j is a transcript descendant of u.

d. For each interval child v_j, the verifier checks that $\gamma(v_j)=\gamma(u)$; that is, if $R(v_j)\neq[0^\ell,1^\ell]$, then $\gamma(v_j)=\gamma(u)$ must hold.

e. For each $j\in[d]$ the verifier checks that v_j is not in a part of the emulation tree that was truncated. (See discussion following Definition 10.) Specifically, for $v\in S(u)$ then if v is a transcript descendant of u, then v_j should not be a transcript descendant of v. For illustration consider Figure 3, where $u,t,z,v\in S$. In that case, the verifier checks that v_j is not a transcript descendant of v (where v is a transcript descendant of u since it was a descendant of u in the original protocol tree).
(Note that it can be that v_j is a transcript descendant of some node in S that is not a transcript descendant of u, and this is not considered a violation. For example, all the nodes are transcript descendants of the root r, which is in S.)

f. The verifier checks that the ranges of all the interval children are disjoint; that is, for every two interval children v_j and v_k, the verifier checks that $R(v_j)\cap R(v_k)=\emptyset$.

g. For each $j\in[d]$ the verifier checks that $\gamma(v_j)$ is **prover consistent** (see Definition 12) with respect to the other transcripts of nodes in S and with regarding to the transcripts of the other children $\gamma(v_k)$, where $k\neq j$.

[5] Because of Step 4, the input node u will always be a non-leaf node.

3. The verifier chooses a child according to the probability distribution J that assigns each $j \in [d]$ probability approximately proportional to $w'(v_j)$ using $O(\log n)$ coin tosses. That is, J is such that

$$\mathbf{Pr}[J = j] \leq \frac{w'(v_j)}{\sum_{i=1}^{d} w'(v_i)} \cdot \left(1 + \frac{1}{n}\right) \tag{5.1}$$

   We can only afford to use $O(\log n)$ public-coins per round, and hence we compromise on sampling each child with probability proportional to $w'(v_j)$, and instead sample with approximate probability. See explanation for approximate sampling in [11, Apdx B].
4. The verifier adds all the children of u to the list S; that is $S \leftarrow S \cup \{v_1, \ldots, v_d\}$. Unless $\gamma(v_j)$ is the complete transcript (which includes the outcome of the coin tosses), the next iteration will start with node v_j and the set S. Otherwise, we proceed to the final checks.

By our conventions, the last message the verifier sends, denoted α_m, contains the outcomes $\rho \in \{0,1\}^{r(n)}$ of the $r(n)$ coins tossed. Thus, if the last node chosen is v, then ρ can be easily extracted from $\gamma(v) = \alpha_1\beta_1, \ldots, \alpha_m\beta_m$. After the last iteration the verifier performs final checks and accepts if all of them hold:

(i) Check that ρ is accepting for $\gamma(v)$ and consistent with it: It checks that $V_0(x, \rho, \beta_1, \ldots, \beta_m) = 1$, and that for every $i = 1, \ldots, m$ it holds that $\alpha_i = V_0(x, \rho, \beta_1, \ldots, \beta_{i-1})$. Note that the verifier needs ρ in order to verify these conditions, so this check can only be done after the last iteration. Also note that if these checks pass then $w(v) = 1$ (rather than $w(v) = 0$).

(ii) Check that $w'(v) = 1$; in other words the prover's last claim should be that the weight of the last node chosen is 1 (and not more than 1).

Clearly, the number of rounds of the emulation is $O(r(n)/\log n)$ because the height of the emulation tree is $O(r(n)/\log n)$, and the prover and verifier proceed one step down the tree on each round. Since the verifier uses $O(\log n)$ public coins on each round, the randomness complexity of the emulation is $O(r(n))$.

References

1 Laszlo Babai. Trading group theory for randomness. In *ACM Symposium on the Theory of Computing*, pages 421–429, 1985.

2 Laszlo Babai and Shlomo Moran. Arthur-merlin games: A randomized proof system and a hierarchy of complexity classes. *Journal of Computer and System Science*, 36:254–276, 1988.

3 Mihir Bellare, Oded Goldreich, and Shafi Goldwasser. Randomness in interactive proofs. *Computational Complexity*, 3:319–354, 1993.

4 Martin Fürer, Oded Goldreich, Yishay Mansour, Michael Sipser, and Stathis Zachos. On completeness and soundness in interactive proof systems. *Advances in Computing Research*, 5:429–442, 1989.

5 Oded Goldreich and Maya Leshkowitz. On emulating interactive proofs with public coins. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:66, 2016.

6 Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. *Journal of the ACM*, 38(3):691–729, 1991. Preliminary version in *27th FOCS*, 1986.

7 Oded Goldreich, Salil P. Vadhan, and Avi Wigderson. On interactive proofs with a laconic prover. *Computational Complexity*, 11(1-2):1–53, 2002.

8 Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989. Preliminary version in *17th STOC*, 1985. Earlier versions date to 1982.

9 Shafi Goldwasser and Michael Sipser. Private coins versus public coins in interactive proof systems. *Advances in Computing Research*, 5:73–90, 1989. Extended abstract in *18th STOC*, 1986.

10 Adam R. Klivans and Dieter van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM Journal on Computing*, 31(5):1501–1526, 2002.

11 Maya Leshkowitz. Round complexity versus randomness complexity in interactive proofs. *Electronic Colloquium on Computational Complexity (ECCC)*, 24:55, 2017. Full version of the paper.

12 Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *Journal of the ACM*, 39(4):859–868, 1992. Extended abstract in *31st FOCS*, 1990.

13 Ronen Shaltiel and Christopher Umans. Low-end uniform hardness versus randomness tradeoffs for AM. *SIAM Journal on Computing*, 39(3):1006–1037, 2009.

14 Adi Shamir. IP = PSPACE. *Journal of the ACM*, 39(4):869–877, 1992. Preliminary version in *31st FOCS*, 1990.

Improved List-Decodability of Random Linear Binary Codes

Ray Li[1]
Department of Computer Science, Stanford University, USA
rayyli@stanford.edu

Mary Wootters
Departments of Computer Science and Electrical Engineering, Stanford University, USA
marykw@stanford.edu

Abstract

There has been a great deal of work establishing that random linear codes are as list-decodable as uniformly random codes, in the sense that a random linear binary code of rate $1 - H(p) - \epsilon$ is $(p, O(1/\epsilon))$-list-decodable with high probability. In this work, we show that such codes are $(p, H(p)/\epsilon + 2)$-list-decodable with high probability, for any $p \in (0, 1/2)$ and $\epsilon > 0$. In addition to improving the constant in known list-size bounds, our argument – which is quite simple – works simultaneously for all values of p, while previous works obtaining $L = O(1/\epsilon)$ patched together different arguments to cover different parameter regimes.

Our approach is to strengthen an existential argument of (Guruswami, Håstad, Sudan and Zuckerman, IEEE Trans. IT, 2002) to hold with high probability. To complement our upper bound for random linear binary codes, we also improve an argument of (Guruswami, Narayanan, IEEE Trans. IT, 2014) to obtain a tight lower bound of $1/\epsilon$ on the list size of uniformly random binary codes; this implies that random linear binary codes are in fact *more* list-decodable than uniformly random binary codes, in the sense that the list sizes are strictly smaller.

To demonstrate the applicability of these techniques, we use them to (a) obtain more information about the distribution of list sizes of random linear binary codes and (b) to prove a similar result for random linear rank-metric codes.

2012 ACM Subject Classification Theory of computation → Error-correcting codes

Keywords and phrases List-decoding, Random linear codes, Rank-metric codes

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.50

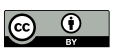
Related Version [26], `https://arxiv.org/abs/1801.07839`

Acknowledgements We thank anonymous reviewers for helpful comments on a previous version of this manuscript.

1 Introduction

An *error correcting code* is a subset $\mathcal{C} \subseteq \mathbb{F}_2^n$, which is ideally "spread out." In this paper, we focus on one notion of "spread out" known as *list-decodability.* We say that a code $\mathcal{C}$ is (p, L)-list-decodable if any Hamming ball of radius pn in $\mathbb{F}_2^n$ contains at most L points of $\mathcal{C}$: that is, if for all $x \in \mathbb{F}_2^n$, $|\mathcal{B}(x, pn) \cap \mathcal{C}| \leq L$, where $\mathcal{B}(x, pn)$ is the Hamming ball of radius pn centered at x. Since list-decoding was introduced in the 1950's [6, 44], it has found

[1] Research supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE - 1656518.



applications beyond communication, for example in pseudorandomness [41] and complexity theory [40].

A classical result in list-decoding is known as the *list-decoding capacity theorem:*

▶ **Theorem 1** (List-decoding capacity theorem). *Let $p \in (0, 1/2)$ and $\varepsilon > 0$.*

1. *There exist binary codes of rate $1 - H(p) - \varepsilon$ that are $(p, \lceil 1/\varepsilon \rceil)$-list-decodable.*
2. *Any binary code of rate $1 - H(p) + \varepsilon$ that is (p, L)-list-decodable up to distance p must have $L \geq 2^{\Omega(\varepsilon n)}$.*

Above, $H(p) = -(1-p)\log_2(1-p) - p\log_2(p)$ is the binary entropy function. We say that a family of binary codes with rate approaching $1 - H(p)$ which are (p, L)-list-decodable for $L = O(1)$ *achieves list-decoding capacity.*[2]

Theorem 1 is remarkable because it means than even when pn is much larger than half the minimum distance of the code – the radius at which at most one codeword $c \in \mathcal{C}$ lies in any Hamming ball – it still can be the case that only a constant number of $c \in \mathcal{C}$ lie in any Hamming ball of radius pn. Because of this, there has been a great deal of work attempting to understand what codes achieve the bound in Theorem 1.

The existential part of Theorem 1 is proved by showing that a uniformly random subset of $\mathbb{F}_2^n$ is $(p, 1/\varepsilon)$-list-decodable with high probability. For a long time, uniformly random codes were the only example of binary codes known to come close to this bound, and today we still do not have many other options. There are explicit constructions of capacity-achieving list-decodable codes over large alphabets (either growing with n or else large-but-constant) [5, 21, 22], but over binary alphabets we still do not have any explicit constructions; we refer the reader to the survey [11] for an overview of progress in this area.

Because it is a major open problem to construct explicit binary codes of rate $1 - H(p) - \varepsilon$ with constant (or even $\text{poly}(n)$) list-sizes, one natural line of work has been to study structured random approaches, in particular *random linear codes.* A random linear code $\mathcal{C} \subset \mathbb{F}_2^n$ is simply a random subspace of $\mathbb{F}_2^n$, and the list-decodability of these codes has been well-studied [45, 13, 12, 2, 43, 34, 36]. There are several reasons to study the list-decodability of random linear codes. Not only is it a natural question in its own right as well as a natural stepping stone in the quest to obtain explicit binary list-decodable codes, but also the list-decodability of random linear codes is useful in other coding-theoretic applications. One example of this is in concatenated codes and related constructions [14, 17, 24, 23], where a random linear code is used as a short inner code. Here, the linearity is useful because (a) a linear code can be efficiently described; (b) it is sometimes desirable to obtain a linear code at the end of the day, hence all components of the construction must be linear; and (c) as in [24] sometimes the linearity is required for the construction to work.

To this end, the line of work mentioned above has aimed to establish that random linear codes are "as list-decodable" as uniformly random codes. That is, uniformly random codes are viewed (as is often the case in coding theory) as the optimal construction, and we try to approximate this optimality with random linear codes, despite the additional structure.

Our contributions

In this paper, we give an improved analysis of the list-decodability of random linear binary codes. More precisely, our contributions are as follows.

[2] Sometimes the phrase "achieves list-decoding capacity" is also used when $L = \text{poly}(n)$; since this paper focuses on the exact constant in the $O(1)$ term however, we use it to mean that $L = O(1)$.

- **A unified analysis.** As we discuss more below, previous work on the list-decodability of random linear binary codes either work only in certain (non-overlapping) parameter regimes [12, 43], or else get substantially sub-optimal bounds on the list-size [36]. Our argument obtains improved list size bounds over all these results and works in all parameter regimes.
  Our approach is surprisingly simple: we adapt an existential argument of Guruswami, Håstad, Sudan and Zuckerman [13] to hold with high probability. Extending the argument in this way was asked as an open question in [13] and had been open until now.
- **Improved list-size for random linear codes.** Not only does our result imply that random linear codes of rate $1-H(p)-\varepsilon$ are (p, L)-list-decodable with list-size $L = O(1/\varepsilon)$, in fact we show that $L \leq H(p)/\varepsilon + 2$. In particular, the leading constant is small and – to the best of our knowledge – is the best known, even existentially, for any list-decodable code.
- **Finer-grained information about the combinatorial structure of random linear codes.** We extend our argument to obtain more information about the distribution of list sizes of random linear codes. More precisely, we obtain high-probability bounds on the number of points x so that the *list size at* x, $L_{\mathcal{C}}(x) := |\mathcal{B}(x, pn) \cap \mathcal{C}|$, is at least ℓ.
- **Tight list-size lower bound for uniformly random codes.** To complement our upper bound, we strengthen an argument of Guruswami and Narayanan [15] to show that a uniformly random binary code of rate $1 - H(p) - \varepsilon$ requires $L \geq (1-\gamma)/\varepsilon$ for any constant $\gamma > 0$ and sufficiently small ε. In other words, the list size of $1/\varepsilon$ in Theorem 1 is tight even in the leading constant. Thus, random linear codes are, with high probability, list-decodable with smaller list sizes than completely random codes.[3]
- **Results for rank-metric codes.** Finally, we adapt our argument for random linear codes to apply to random linear rank-metric codes. As with standard (Hamming-metric) codes, recent work aimed to show that random linear rank-metric codes are nearly as list-decodable as uniformly random codes [4, 16]. Our approach establishes that in fact, random linear binary rank-metric codes are more list-decodable than their uniformly random counterparts in certain parameter regimes, in the sense that the list sizes near capacity are strictly smaller. Along the way, we show that low-rate random linear binary rank-metric codes are list-decodable to capacity, answering a question of [16].

On the downside, we note that our arguments only work for binary codes and do not extend to larger alphabets; additionally, our positive results do not establish *average-radius* list-decodability with list size $O(1/\varepsilon)$, a stronger notion which was established in some of the previous works [2, 43, 36]. It would be very interesting to extend our results to these settings.

1.1 Outline of paper

After a brief overview of the notation in §1.2, we proceed in §2 with a survey of related work for both random linear codes and rank-metric codes, and we formally state our results in this context. In §3, we prove Theorem 5, which establishes our upper bound for random linear binary codes. In Appendix A, we prove Theorem 6, characterizing the list size distribution of random linear codes. We refer the interested reader to the full version of the paper [26] for proofs of our other results, Theorems 7, 11, and 12.

[3] In retrospect, this may not be surprising: for example, it is well-known that random linear codes have better distance than completely random codes. However, the fact that we are able to prove this is surprising to the authors, since it requires taking advantage of the dependence between the codewords, rather than trying to get around it.

1.2 Notation

Throughout most of the paper, we are interested in binary codes $\mathcal{C} \subseteq \mathbb{F}_2^n$ of *block length* n. The *dimension* of a code $\mathcal{C}$ is defined as $k = \log_2 |\mathcal{C}|$, and the *rate* is the ratio k/n. We define a *uniformly random binary code of rate* R to be a set $\mathcal{C}$ of 2^{Rn} elements chosen independently and uniformly at random from $\mathbb{F}_2^n$. We say that a binary code is *linear* if it forms a linear subspace of $\mathbb{F}_2^n$. We define a *random linear binary code of rate* R to be the span of $k = Rn$ independently random vectors $b_1, \ldots, b_k \in \mathbb{F}_2^n$.[4]

For two points $x, y \in \mathbb{F}_2^n$, we use $\Delta(x, y) = \sum_{i=1}^n \mathbb{I}[x_i = y_i]$ to denote the Hamming distance between x and y, where, for an event $\mathcal{E}$, $\mathbb{I}[\mathcal{E}]$ is 1 if $\mathcal{E}$ occurs and 0 otherwise. For $x \in \mathbb{F}_2^n, r \in [0, n]$, we define the Hamming ball $\mathcal{B}(x, r)$ of radius r centered at x to be $\mathcal{B}(x, r) = \{y \in \mathbb{F}_2^n : \Delta(x, y) \leq r\}$, and the volume of $\mathcal{B}(x, r)$ to be $\mathrm{Vol}(n, r) := |\mathcal{B}(0^n, r)| = \sum_{i=0}^r \binom{n}{i}$. We use the well known bound that, for any $p \in [0, 1]$, $\mathrm{Vol}(n, pn) \leq 2^{H(p)n}$, where $H(p) = -(1-p)\log_2(1-p) - p\log_2(p)$ is the binary entropy function. One of our main technical results is about the distribution of *list sizes* of points $x \in \mathbb{F}_2^n$: given a code $\mathcal{C}$ and $p \in (0, 1/2)$, we define the list size of a point $x \in \mathbb{F}_2^n$ to be $L_{\mathcal{C}}(x) := |\mathcal{B}(x, pn) \cap \mathcal{C}|$.

For $\alpha > 0, \beta \in \mathbb{R}$, let $\exp_\alpha(\beta) := \alpha^\beta$ and assume $\alpha = e$ when it is omitted. For two sets $A, B \subseteq \mathbb{F}_2^n$, define the sumset $A + B = \{a + b : a \in A, b \in B\}$. When $b \in \mathbb{F}_2^n$, let $A + b$ denote $A + \{b\}$.

2 Previous Work and Our Results

In §2.1 below, we survey related work on the list-decodability of random linear binary codes. In §2.2, we state our positive results for random linear codes. In §2.3, we state our negative result for uniformly random codes. In §2.4, we introduce and survey existing work on random rank-metric codes. In §2.5, we state our results on the list-decodability of random linear and uniformly random binary rank-metric codes.

2.1 Prior work: uniformly random and random linear codes

The list-decodability of random linear binary codes has been well studied. Here we survey the results that are most relevant for this work. As this work focuses on binary codes, we focus this survey on results for binary codes, even though many of the works mentioned also apply to general q-ary codes. We additionally remark that, in contrast to the large alphabet setting [18], capacity achieving binary codes have no known explicit constructions.

A modification of the proof of the list-decoding capacity theorem shows that a random linear code of rate $1 - H(p) - \varepsilon$ is $(p, \exp(O(\frac{1}{\varepsilon})))$-list-decodable [45]. However, whether or not random linear codes of this rate have list-sizes that do not depend exponentially on ε remained open for decades: this question was explicitly asked in [7].

A first step was given in the work of Guruswami, Håstad, Sudan and Zuckerman [13], who proved via a beautiful potential-function-based-argument that there *exist* binary linear codes or rate $1 - H(p) - \varepsilon$ which are $(p, 1/\varepsilon)$-list-decodable. However, this result did not hold with high probability. Our approach relies heavily on the approach of [13], and we return to their argument in §3.

[4] Our definitions of uniformly random binary code and random linear binary code are slightly different than the standard definitions, which are a uniformly random set of size 2^{Rn} and a uniformly random subspace of dimension k, respectively. However, our definitions are easier to work with. Furthermore, for this paper, the difference is negligible. For example, a random linear code has rank strictly less than k with probability at most $2^{-(n-k)}$, and each dimension k code is represented in the same number of ways, so our results hold also for the more standard definition.

Over the next 15 years, a line of work [12, 2, 43, 34, 35, 36] has focused on the list-decodability (and related properties) of random linear codes, which should hold with high probability. The works most relevant to ours are [12, 43], which together more or less settle the question. We state these results here for binary alphabets, although both works address larger alphabets as well.

The first result, of [12], establishes a result for a constant p, bounded away from $1/2$.

▶ **Theorem 2** (Theorem 2 of [12]). *Let $p \in (0, 1/2)$. Then there exist constants $C_p, \delta > 0$ such that for all $\varepsilon > 0$ and sufficiently large n, for all $R \leq 1 - H(p) - \varepsilon$, if $\mathcal{C} \subseteq \mathbb{F}_2^n$ is a random linear code of rate R, then $\mathcal{C}$ is $(p, C_p/\varepsilon)$-list-decodable with probability at least $1 - 2^{-\delta n}$.*

However, C_p is not small and tends to ∞ as p approaches $1/2$. The following result of [43] fills in the gap when p is quite close to $1/2$.

▶ **Theorem 3** (Theorem 2 of [43]). *There exist constants C_1, C_2 so that for all sufficiently small $\varepsilon > 0$ and sufficiently large n, for $p = 1/2 - C_1\sqrt{\varepsilon}$ and for all $R \leq 1 - H(p) - \varepsilon$, if $\mathcal{C} \subseteq \mathbb{F}_2^n$ is a random linear code of rate R, then $\mathcal{C}$ is $(p, C_2/\varepsilon)$-list-decodable with probability at least $1 - o(1)$.*

The list-decoding capacity theorem implies that we cannot hope to take the rate R substantially larger than $1 - H(p) - \varepsilon$ and obtain a constant list size. Moreover, the list size $\Theta(1/\varepsilon)$ is optimal for both random linear codes and uniformly random codes [32, 15]. More precisely, Guruswami and Narayanan show the following theorem (which we have specialized to binary codes).

▶ **Theorem 4** (Theorem 20 of [15]). *Let $\varepsilon > 0$. A uniformly random binary code of rate $1 - H(p) - \varepsilon$ is $(p, (1 - H(p))/\varepsilon)$-list-decodable with probability at most $\exp(-\Omega_{p,\varepsilon}(n))$.*[5]

We note that for general codes (not uniformly random or random linear) it is still unknown what the "correct" list size L is in terms of ε, although there are results in particular parameter regimes [1, 19] and for stronger notions of list-decodability [15].

2.2 Our main results: random linear codes

We show that, with high probability, a random linear binary code of rate $1 - H(p) - \varepsilon$ is (p, L)-list-decodable with $L \sim H(p)/\varepsilon$. More precisely, the upper bound is as follows (proved in §3).

▶ **Theorem 5.** *Let $p \in (0, 1/2)$, let $\varepsilon > 0$, and let $R = 1 - H(p) - \varepsilon$. Let $\mathcal{C} \subseteq \mathbb{F}_2^n$ be a random linear code of rate R. Then with probability $1 - \exp(-\Omega_\varepsilon(n))$, the code $\mathcal{C}$ is $(p, H(p)/\varepsilon + 2)$-list-decodable.*

Theorem 5 improves upon the picture given by Theorems 2 and 3 in two ways. First, the leading constant on the list size, which is $H(p)$, improves over both the constant C_p from Theorem 2 (which blows up as $p \to 1/2$) and on the constant C_2 from Theorem 3 (which the authors do not see how to make less than 2). Moreover, when $p \to 1/2$, Theorem 5 improves on Theorem 3 in that it decouples p from ε: in Theorem 3, we must take $p = 1/2 - O(\sqrt{\varepsilon})$

[5] In fact, in [15], Theorem 20 is stated with a list size of $(1 - H(p))/2\varepsilon$ as a lower bound. However, the constant can be improved to $1 - H(p)$, because the factor of 2 is introduced to handle an additive constant term. Thus, for sufficiently large n their argument proves the stronger statement stated above.

and $R = 1 - H(p) - \varepsilon$, while in Theorem 5, p and ε may be chosen independently. Thus, Theorem 5 offers the first true "list-decoding capacity theorem for binary linear codes," in that it precisely mirrors the quantifiers in Theorem 1.

The techniques that we use to prove Theorem 5 can be refined to give more combinatorial information about random linear codes. It is our hope that such information will help in further derandomizing constructions of binary codes approaching list-decoding capacity. In Appendix A, we prove the following structural result about random linear binary codes.

▶ **Theorem 6.** *Let $\varepsilon, \gamma \in (0,1)$ be constants, $p \in (0,1/2)$, L be a positive integer, and let $R = 1 - H(p) - \varepsilon$. Let $\mathcal{C} \subset \mathbb{F}_2^n$ be a random linear code of rate R. Then with probability $1 - \exp(-\Omega_L(\gamma\varepsilon n))$, the code $\mathcal{C}$ satisfies, for all $0 \le \ell \le L$,*

$$|\{x \in \mathbb{F}_2^n : L_{\mathcal{C}}(x) \ge \ell\}| \;\le\; 2^n \cdot 2^{-n\ell\varepsilon(1-\gamma)}.$$

To interpret this result, it is helpful to think of γ as close to 0. Intuitively, it says that, with high probability over the choice of a random linear code $\mathcal{C}$, the number of words $x \in \mathbb{F}_2^n$ with "list size ℓ" decays approximately exponentially as $2^{-n\ell\varepsilon}$. As we show in Appendix A, Theorem 5 follows as a corollary of Theorem 6 (see Corollary 19), but as a warm-up to Theorem 6 we present a proof of Theorem 5 independent of Theorem 6 in §3.

▶ Remark. Theorem 6 implies that, with high probability over the choice of the code, for any codeword $c \in \mathcal{C}$, $\mathbf{Pr}_{x \in \mathcal{B}(c,pn)}[L_{\mathcal{C}}(x) = 1] \ge 1 - 2^{-n(\varepsilon(1-\gamma)-o(1))}$. That is, "most" points $x \in \{0,1\}^n$ within pn of a codeword $c \in \mathcal{C}$ are not within pn of any other codeword $c' \ne c$. This is in line with the conventional wisdom from the Shannon model: with high probability, random linear codes achieve capacity on the BSC, so for a random linear code, a random center x obtained by sending a codeword $c \in \mathcal{C}$ through the binary symmetric channel $\mathrm{BSC}(p)$ has $L_{\mathcal{C}}(x) \le 1$ with high probability. (See also [33]). Thus, Theorem 6 recovers this intuition for list size 1, and quantitatively extends it to list sizes larger than 1.

2.3 Our results: uniformly random codes

In Theorem 5, the list size of $H(p)/\varepsilon + 2$ is smaller than the list size of $1/\varepsilon$ given by the classical list-decoding capacity theorem for uniformly random codes. Further, the following negative result shows that the list size of $1/\varepsilon$ given by uniformly random binary codes in the list-decoding capacity theorem is tight, even in the leading constant of 1.

▶ **Theorem 7.** *For any $p \in (0,1/2)$ and $\varepsilon > 0$, there exists a $\gamma_{p,\varepsilon} = \exp(-\Omega_p(\frac{1}{\varepsilon}))$ and $n_{p,\varepsilon} \in \mathbb{N}$ such that for all $n \ge n_{p,\varepsilon}$, a random linear code $\mathcal{C} \subseteq \mathbb{F}_2^n$ of rate $R = 1 - H(p) - \varepsilon$ is with probability $1 - \exp(-\Omega_{p,\varepsilon}(n))$ not $(p, \frac{1-\gamma_{p,\varepsilon}}{\varepsilon})$-list-decodable.*

The proof of Theorem 7 is obtained by tightening the second moment method proof of [15], and can be found in Appendix A of [26]. Theorem 7, combined with Theorem 5, implies that, for all $p \in (0,1/2)$ and sufficiently small ε, random linear codes of rate $1 - H(p) - \varepsilon$ with high probability can be list-decoded up to distance p with smaller list sizes than uniformly random codes. Perhaps surprisingly, the difference between the list size upper bound in Theorem 1 and the lower bound in Theorem 7 is bounded by 1 as $\varepsilon \to 0$, implying that the "correct" list size of a uniformly random code is tightly concentrated between $\lfloor 1/\varepsilon \rfloor \pm 1$ for small ε.

We are unaware of results in the literature that give even the existence of binary codes list-decodable with list size *better* than $H(p)/\varepsilon$. We remark that the Lovasz Local Lemma also gives the *existence* of $(p, H(p)/\varepsilon)$-list-decodable codes, matching our high probability result for random linear codes. We refer the reader to Appendix C of [26] for the details.

2.4 Prior work: rank metric codes

As an application of our techniques for random linear codes, we turn our attention to *rank metric codes*. Rank metric codes, introduced by Delsarte in [3], are codes $\mathcal{C} \subseteq \mathbb{F}_q^{m\times n}$; that is, the codewords are $m \times n$ matrices, where typically $m \geq n$. The distance between two codewords X and Y is given by the rank of their difference: $\Delta_R(X,Y) := \frac{1}{n}\operatorname{rank}(X-Y)$, where Δ_R is called the *rank metric*. We denote the *rank ball* by $\mathcal{B}_{q,R}(X,p) := \{Y \in \mathbb{F}_q^{m\times n} : \Delta_R(X<Y) \leq pn\}$, and say that a rank metric code $\mathcal{C} \subseteq \mathbb{F}_q^{m\times n}$ is (p,L)-list-decodable if $|\mathcal{B}_{q,R}(X,p) \cap \mathcal{C}| \leq L$ for all $X \in \mathbb{F}_q^{m\times n}$. The *rate* R of a rank metric code $\mathcal{C} \subset \mathbb{F}_q^{m\times n}$ is $R := \log_q(|\mathcal{C}|)/(mn)$.

Rank metric codes generalize standard (Hamming metric) codes, which are simply diagonal rank metric codes. The study of rank metric codes has been motivated by a wide range of applications, including magnetic storage [31], cryptography [8, 27, 28], space-time coding [30, 29], and network coding [25, 39], and distributed storage [38, 37].

The natural "list-decoding capacity" for rank metric codes is $R = (1-p)(1-(n/m)p)$, which is the analog of the Gilbert-Varshamov bound [10]. It was shown in [4, 16] that this is achievable by a uniformly random rank metric code.

▶ **Theorem 8** ([16], Proposition A.1.[6]). *Let $\varepsilon > 0$ and $p \in (0,1)$ and suppose that m, n are sufficiently large compared to $1/\varepsilon$. A uniformly random code $\mathcal{C} \subseteq \mathbb{F}_q^{m\times n}$ of rate $R = (1-p)(1-bp) - \varepsilon$ is $(p, \lceil 1/\varepsilon \rceil)$-list-decodable with probability at least $1 - O(q^{-\varepsilon mn})$, where $b = n/m$.*

Moreover, it is shown in [4] that no linear code can beat this bound.

▶ **Theorem 9.** *Let $b = \lim_{n\to\infty} \frac{n}{m}$ be a constant. Then for any $R \in (0,1)$ and $p \in (0,1)$, a (p,L)-list-decodable $\mathbb{F}_q$-linear rank metric code $\mathcal{C} \subseteq \mathbb{F}_q^{m\times n}$ with rate R satisfies $R \leq (1-p)(1-bp)$.*

There has been a great deal of work aimed at establishing (or refuting) the list-decodability of explicit rank metric codes. It is shown in [42] that Gabidulin codes [9] – the rank-metric analog of Reed-Solomon codes – are *not* list-decodable to capacity, or even much beyond half their minimum distance. However, there have been works [22, 20] designing explicit codes meeting the lower bound of Theorem 9.

As with standard (Hamming-metric) codes, it is interesting to study the list-decodability of random linear rank-metric codes; we say that $\mathcal{C} \subseteq \mathbb{F}_q^{n\times m}$ is linear if it forms an $\mathbb{F}_q$-linear space. Following the approach of [45] for Hamming metric codes, [4] shows that random linear rank metric codes of rate $R = (1-p)(1-bp) - \varepsilon$ are $(p, \exp(O(1/\varepsilon))$-list-decodable, where as above $b = n/m$. In a recent paper of Guruswami and Resch [16], this result was strengthened to give a list size of $O(1/\varepsilon)$.

▶ **Theorem 10** ([16]). *Let $p \in (0,1)$ and $q \geq 2$. There is some constant $C_{p,q}$ so that the following holds. For all sufficiently large n, m with $b = n/m$, a random linear rank metric code $\mathcal{C} \subseteq \mathbb{F}_q^{m\times n}$ of rate $R = (1-p)(1-bp) - \varepsilon$ is $(p, C_{p,q}/\varepsilon)$-list-decodable with high probability.*

The proof of Theorem 10 uses ideas from the approach of [12] to prove Theorem 2. This is a beautiful argument, but as with the results of [12], the result of [16] suffers from the fact that $C_{p,q}$ tends to ∞ as p approaches 1. It is shown in [16], Proposition A.2, that

[6] In [16], the result is stated with $L = O(1/\varepsilon)$, but an inspection of the proof shows that we may take the leading constant to be 1.

when $p = 1 - \eta$, a uniformly random rank metric code of rate $R = (\eta - \eta b + \eta^2 b)/2$ is $(p, 4/(\eta - \eta b + \eta^2 b))$-list-decodable, and that work poses the question of whether or not a random linear rank metric code can achieve this. Our results, described in the next section, show that the answer is "yes" for $q = 2$.

2.5 Our results: rank metric codes

By applying the techniques in the proof of Theorem 5, we prove the following upper bound on the list size of random linear binary rank-metric codes.

▶ **Theorem 11.** *Let $p \in (0,1)$ and $\varepsilon > 0$. There is a constant C_ε so that the following holds. Let m and n be sufficiently large positive integers with $n < m$ and let $b = n/m$. A random linear rank metric code $\mathcal{C} \subseteq \mathbb{F}_2^{m \times n}$ of rate $R = (1-p)(1-bp) - \varepsilon$ is $(p, \frac{p+bp-bp^2}{\varepsilon} + 2)$-list-decodable with probability at least $1 - \exp(-C_\varepsilon mn)$.*

Notice that Theorem 11 works for all p, improving upon Theorem 10. In particular, when $p = 1 - \eta$, then setting $\varepsilon = (1-p)(1-bp)/2$ and applying Theorem 11 implies that a random linear binary rank metric code of rate $R = (\eta - \eta b + \eta^2 b)/2$ is (p, L) list-decodable for $L \leq \frac{2}{\eta - \eta b + \eta^2 b}$, answering the aforementioned open question of [16] in the affirmative.

We also prove a new lower bound on the list size of uniformly random rank-metric codes.

▶ **Theorem 12.** *Let $p \in (0,1)$ and $\varepsilon > 0$. Suppose m, n are sufficiently large so that $b = n/m$. Let $\mathcal{C} \subseteq \mathbb{F}_q^{m \times n}$ be a uniformly random rank metric code of rate $R = (1-p)(1-bp) - \varepsilon$. Then $\mathcal{C}$ is $(p, (1-p)(1-bp)/\varepsilon - 1)$-list-decodable with probability at most $\exp(-\Omega_{p,\varepsilon}(n))$.*

Theorem 12 again uses the method of [15]. The proofs of Theorems 11 and 12 can be found in Section 5 and Appendix B, respectively, of [26]. Together, Theorems 11 and 12 show that for some values of p, random linear binary rank metric codes have a strictly smaller list size than uniformly random rank metric codes with the same parameters. In particular, the upper bound of Theorem 11 is strictly smaller than the lower bound of Theorem 12 whenever $p < \frac{1-b}{2}$. For larger values of p, we remark that the list size obtained by Theorem 11 is still strictly smaller than the $1/\varepsilon$ list size given by uniformly random codes in Theorem 8, even though in this case we don't have a lower bound which proves that this is tight.

3 Simplified result for random linear binary codes

In this section, we prove Theorem 5, which we restate here.

▶ **Theorem 13** (Theorem 5, restated)**.** *Let $p \in (0, 1/2)$, let $\varepsilon > 0$, and let $R = 1 - H(p) - \varepsilon$. Let $\mathcal{C} \subseteq \mathbb{F}_2^n$ be a random linear code of rate R. Then with probability $1 - \exp(-\Omega_\varepsilon(n))$, the code $\mathcal{C}$ is $(p, H(p)/\varepsilon + 2)$-list-decodable.*

Theorem 5 also follows from our more refined result, Theorem 6. However, since our techniques give a very simple proof of Theorem 5 on its own, we begin with just this simple proof. We start by reviewing the approach of [13], which is the basis of our proof.

3.1 The approach of [13]

Before anything was known about the list-decodability of a typical random linear code, Guruswami, Håstad, Sudan and Zuckerman [13] proved the *existence* of binary linear codes of rate $1 - H(p) - \varepsilon$ that are $(p, 1/\varepsilon)$-list-decodable. Their result followed from a beautiful potential-function argument, which is the basis of our approach and which we describe here.

Let $k := Rn = (1 - H(p) - \varepsilon)n$. We choose vectors $b_1, \ldots, b_k$ one at a time, so that the code $\mathcal{C}_i := \text{span}(b_1, \ldots, b_i)$ remains "nice": formally, so that a potential function $\tilde{S}_{\mathcal{C}_i}$ remains small. Once we have picked all k vectors, we set $\mathcal{C} = \mathcal{C}_k$, and the fact that $\tilde{S}_{\mathcal{C}_k}$ is small implies list-decodability.

Recall that for a code $\mathcal{C}$ and $x \in \mathbb{F}_2^n$, we set $L_{\mathcal{C}}(x) = |\mathcal{B}(x, pn) \cap \mathcal{C}|$. Define

$$\tilde{S}_{\mathcal{C}} := \frac{1}{2^n} \sum_{x \in \mathbb{F}_2^n} 2^{\varepsilon n L_{\mathcal{C}}(x)}.$$

It is not hard to show that for any vectors $b_1, \ldots, b_i \in \mathbb{F}_2^n$,

$$\mathop{\mathbf{E}}_{b_{i+1} \sim \mathbb{F}_2^n} \left[\tilde{S}_{\mathcal{C}_i + \{0, b_{i+1}\}} | b_1, \ldots, b_i \right] \leq \tilde{S}_{\mathcal{C}_i}^2. \tag{1}$$

That is, when a uniformly random vector b_{i+1} is added to the basis $\{b_1, \ldots, b_i\}$, we expect the potential function not to grow too much. Hence, there exists a choice of vectors $b_1, \ldots, b_k$ so that $\tilde{S}_{\mathcal{C}_{i+1}} \leq \tilde{S}_{\mathcal{C}_i}^2$ for $i = 0, 1, \ldots, k-1$.[7]

As $\mathcal{C}_0 = \{0\}$, we have $\tilde{S}_{\mathcal{C}_0} \leq 1 + 2^{-n(1-H(p)-\varepsilon)}$. Setting $\mathcal{C} = \mathcal{C}_k = \text{span}(b_1, \ldots, b_k)$, we have

$$\tilde{S}_{\mathcal{C}} \leq \tilde{S}_{\mathcal{C}_0}^{2^k} \leq \left(1 + 2^{-n(1-H(p)-\varepsilon)}\right)^{2^k} \leq \exp\left(2^{k-n(1-H(p)-\varepsilon)}\right) \leq e$$

by our choice of k. This implies that $\sum_x 2^{\varepsilon n L_{\mathcal{C}}(x)} \leq e \cdot 2^n$, and in particular, for all $x \in \mathbb{F}_2^n$, we have $2^{\varepsilon n L_{\mathcal{C}}(x)} \leq e \cdot 2^n$. Thus, for all x, $L_{\mathcal{C}}(x) \leq \frac{1}{\varepsilon} + o(1)$, as desired.

The approach of [13] is extremely clever, but these ideas have not, to the best of our knowledge, been used in subsequent work on the list-decodability of random linear codes. One reason is that the crux of the argument, which is (1), holds in expectation, and it was not clear how to show that it holds with high probability; thus, the result remained existential, and other techniques were introduced to study typical random codes [12, 2, 43, 34, 36].

3.2 Proof of Theorem 5

We improve the argument of [13] in two ways. First, we show that in fact, (1) essentially holds with high probability over the choice of b_{i+1}, which allows us to use the approach sketched above for random linear codes. Second, we introduce one additional trick which takes advantage of the linearity of the code in order to reduce the constant in the list size from 1 to $H(p)$. Before diving into the details, we briefly describe the main ideas.

The first improvement follows from looking at the potential function in the right way. In this paragraph, all $o(1)$ terms are exponentially small in n. Our goal is $\tilde{S}_{\mathcal{C}_k} \leq O(1)$. Write $\tilde{S}_{\mathcal{C}_i} = 1 + \tilde{T}_{\mathcal{C}_i}$. By above, $\tilde{T}_{\mathcal{C}_0} = \tilde{S}_{\mathcal{C}_0} - 1 = o(1)$. We show that with high probability, for all $i \leq k$, we have $\tilde{T}_{\mathcal{C}_i} = o(1)$. In the [13] argument we have

$$\mathbf{E}\, \tilde{S}_{\mathcal{C}_{i+1}} \leq \tilde{S}_{\mathcal{C}_i}^2 = (1 + \tilde{T}_{\mathcal{C}_i})^2 = 1 + 2\tilde{T}_{\mathcal{C}_i}(1 + o(1)),$$

and so $\mathbf{E}\, \tilde{T}_{\mathcal{C}_{i+1}} = 2\tilde{T}_{\mathcal{C}_i}(1 + o(1))$. One can show that, always, $2\tilde{T}_{\mathcal{C}_i} \leq \tilde{T}_{\mathcal{C}_{i+1}}$. By Markov's inequality, $\tilde{T}_{\mathcal{C}_{i+1}} = 2\tilde{T}_{\mathcal{C}_i}(1 + o(1))$ with probability $1 - o(1)$, for appropriately chosen $o(1)$ terms. Union bounding over the $o(1)$ failure probabilities in the k steps, we conclude that $\tilde{T}_{\mathcal{C}_i}$ grows roughly as slowly as in the existential argument, giving the desired list-decodability.

[7] As a technical detail, one needs to be careful that $b_{i+1} \notin \mathcal{C}_i$. One can guarantee $b_{i+1} \notin \mathcal{C}_i$ by carefully examining the proof of (1), or use (1) to get a similar equation where we additionally condition $b_{i+1} \notin \mathcal{C}_i$.

The second improvement follows from the linearity of the code. In the last step of the [13] argument, we replace the summation "$\sum_x$" in $\sum_x 2^{\varepsilon n L_{\mathcal{C}}(x)} \le e \cdot 2^n$ with a "$\forall x$." We can save a bit because, by linearity, the contribution $2^{\varepsilon n L_{\mathcal{C}}(x)}$ is the same for all x in a coset $y + \mathcal{C}$.

Now we go through the details. It is convenient to change the definition of the potential function very slightly: losing the tilde, define, for a code $\mathcal{C} \subset \mathbb{F}_2^n$,

$$A_{\mathcal{C}}(x) \;:=\; 2^{\frac{\varepsilon n L_{\mathcal{C}}(x)}{1+\varepsilon}} \qquad \text{and} \qquad S_{\mathcal{C}} \;:=\; \mathop{\mathbf{E}}_{x \sim \mathbb{F}_2^n} \left[A_{\mathcal{C}}(x)\right].$$

The term $S_{\mathcal{C}}$ differs from the term $\tilde{S}_{\mathcal{C}}$ above in that $A_{\mathcal{C}}(x)$ has an extra factor of $\frac{1}{1+\varepsilon}$ in the exponent. This is an extra "slack" term that helps guarantee a high probability result under the same parameters. However, this definition does not change how the potential function behaves. In particular, we still have the following lemma:

▶ **Lemma 14** (Following [13]). *For all linear $\mathcal{C} \subseteq \mathbb{F}_2^n$ and all $b \in \mathbb{F}_2^n$,*

$$L_{\mathcal{C}+\{0,b\}}(x) \;\le\; L_{\mathcal{C}}(x) + L_{\mathcal{C}}(x+b) \tag{2}$$
$$A_{\mathcal{C}+\{0,b\}}(x) \;\le\; A_{\mathcal{C}}(x) \cdot A_{\mathcal{C}}(x+b), \tag{3}$$

with equality if and only if $b \notin \mathcal{C}$.

Proof. To see (2), notice that

$$\begin{aligned} L_{\mathcal{C}+\{0,b\}}(x) \;&=\; |\mathcal{B}(x, pn) \cap (\mathcal{C} \cup (\mathcal{C}+b))| \\ &\le\; |\mathcal{B}(x, pn) \cap \mathcal{C}| + |\mathcal{B}(x, pn) \cap (\mathcal{C}+b)| \\ &=\; |\mathcal{B}(x, pn) \cap \mathcal{C}| + |\mathcal{B}(x+b, pn) \cap \mathcal{C}| \\ &=\; L_{\mathcal{C}}(x) + L_{\mathcal{C}}(x+b), \end{aligned}$$

with equality in the second line if and only if $b \notin \mathcal{C}$. Inequality (3) follows as a consequence of (2), and this proves the lemma. ◀

We additionally define

$$B_{\mathcal{C}}(x) \;:=\; A_{\mathcal{C}}(x) - 1 \qquad \text{and} \qquad T_{\mathcal{C}} \;:=\; S_{\mathcal{C}} - 1.$$

As noted above, it is helpful to think of $T_{\mathcal{C}}$ as a very small term; we would like to show – in accordance with (1) – that $T_{\mathcal{C}}$ approximately doubles each time we add a basis vector. Note that

$$S_{\{0\}} \;=\; 1 + \left(2^{\frac{\varepsilon n}{1+\varepsilon}} - 1\right) \cdot \frac{\mathrm{Vol}(n, pn)}{2^n} \;<\; 1 + 2^{\frac{\varepsilon n}{1+\varepsilon}} \cdot \frac{2^{H(p)n}}{2^n} \;=\; 1 + 2^{-n(1-H(p)-\frac{\varepsilon}{1+\varepsilon})}. \tag{4}$$

With these definitions, we can prove the concentration result we need.

▶ **Lemma 15.** *Let p, ε, and $R = 1 - H(p) - \varepsilon$ be as in the statement of Theorem 5. Let $\mathcal{C}_{Rn} \subset \mathbb{F}_2^n$ be a random linear code of rate R. Then with probability $1 - \exp(-\Omega_\varepsilon(n))$, the code $\mathcal{C}_{Rn}$ satisfies $S_{\mathcal{C}_{Rn}} \le 2$.*

Before we prove Lemma 15 we show that this implies Theorem 5.

Proof of Theorem 5 given Lemma 15. We show that, for a binary linear code $\mathcal{C}$ of rate $R = 1 - H(p) - \varepsilon$, $S_{\mathcal{C}} \le 2$ implies $(p, \frac{H(p)}{\varepsilon} + 2)$-list-decodability. Suppose for sake of contradiction

that a code $\mathcal{C}$ satisfies $S_{\mathcal{C}} \leq 2$ and there exists $x^* \in \mathbb{F}_2^n$ such that $|\mathcal{B}(x^*, pn) \cap \mathcal{C}| > H(p)/\varepsilon + 2$. For all $x \in \mathbb{F}_2^n$ and $c \in \mathcal{C}$, we have

$$|\mathcal{B}(x+c, pn) \cap \mathcal{C}| = |\mathcal{B}(x, pn) \cap (\mathcal{C} - c)| = |\mathcal{B}(x, pn) \cap \mathcal{C}|,$$

so $|\mathcal{B}(x^* + c, pn) \cap \mathcal{C}| > H(p)/\varepsilon$ for all $c \in \mathcal{C}$. If $S_{\mathcal{C}} \leq 2$, then we have

$$\begin{aligned}
2^{n+1} \geq 2^n S_{\mathcal{C}} &= \sum_{x \in \mathbb{F}_2^n} \exp_2\left(n \cdot \frac{\varepsilon}{1+\varepsilon} \cdot |\mathcal{B}(x, pn) \cap \mathcal{C}|\right) \\
&> \sum_{c \in \mathcal{C}} \exp_2\left(n \cdot \frac{\varepsilon}{1+\varepsilon} \cdot |\mathcal{B}(x^* + c, pn) \cap \mathcal{C}|\right) \\
&\geq \sum_{c \in \mathcal{C}} \exp_2\left(n \cdot \frac{\varepsilon}{1+\varepsilon} \cdot (H(p)/\varepsilon + 2)\right) \\
&= |\mathcal{C}| \cdot \exp_2\left(n \cdot \frac{H(p) + 2\varepsilon}{1+\varepsilon}\right) \\
&= \exp_2\left(n\left(1 + \frac{\varepsilon}{1+\varepsilon}(1 - H(p) - \varepsilon)\right)\right),
\end{aligned}$$

where the first inequality is because we sum over strictly fewer terms, and the second inequality is by definition of x^*. This is a contradiction for large enough n. ◀

Finally, we prove Lemma 15.

Proof of Lemma 15. As in §3.1, let $b_1, b_2, \ldots, b_k \in \mathbb{F}_2^n$ be independently and uniformly chosen, and let $\mathcal{C}_i = \operatorname{span}\{b_1, \ldots, b_i\}$.

▶ **Lemma 16.** *Suppose that $\mathcal{C}$ is fixed and satisfies $T_{\mathcal{C}} < 1$, so that $S_{\mathcal{C}} < 2$. Then*

$$\Pr_{b \sim \mathbb{F}_2^n}\left[S_{\mathcal{C}+\{0,b\}} > 1 + 2T_{\mathcal{C}} + T_{\mathcal{C}}^{1.5}\right] < T_{\mathcal{C}}^{0.5}.$$

Proof. By Lemma 14, for all b,

$$\begin{aligned}
S_{\mathcal{C}+\{0,b\}} &= \mathop{\mathbf{E}}_x\left[A_{\mathcal{C}+\{0,b\}}(x)\right] \\
&\leq \mathop{\mathbf{E}}_x\left[A_{\mathcal{C}}(x) A_{\mathcal{C}}(x+b)\right] \\
&= \mathop{\mathbf{E}}_x\left[(1 + B_{\mathcal{C}}(x)) \cdot (1 + B_{\mathcal{C}}(x+b))\right] \\
&= \mathop{\mathbf{E}}_x\left[1 + B_{\mathcal{C}}(x) + B_{\mathcal{C}}(x+b) + B_{\mathcal{C}}(x) B_{\mathcal{C}}(x+b)\right] \\
&= 1 + 2T_{\mathcal{C}} + \mathop{\mathbf{E}}_x\left[B_{\mathcal{C}}(x) B_{\mathcal{C}}(x+b)\right].
\end{aligned}$$

Over the randomness of b and x, we have x and $x+b$ are statistically independent and uniform over $\mathbb{F}_2^n$, so we have

$$\mathop{\mathbf{E}}_b \mathop{\mathbf{E}}_x\left[B_{\mathcal{C}}(x) B_{\mathcal{C}}(x+b)\right] = \mathop{\mathbf{E}}_{b,x}\left[B_{\mathcal{C}}(x)\right] \cdot \mathop{\mathbf{E}}_{b,x}\left[B_{\mathcal{C}}(x+b)\right] = T_{\mathcal{C}}^2. \tag{5}$$

As $B_{\mathcal{C}}$ is always nonnegative, we have, by Markov's inequality,

$$\Pr_b\left[S_{\mathcal{C}+\{0,b\}} > 1 + 2T_{\mathcal{C}} + T_{\mathcal{C}}^{1.5}\right] \leq \Pr_b\left[\mathop{\mathbf{E}}_x\left[B_{\mathcal{C}}(x) B_{\mathcal{C}}(x+b)\right] > T_{\mathcal{C}}^{1.5}\right] < \frac{T_{\mathcal{C}}^2}{T_{\mathcal{C}}^{1.5}} = T_{\mathcal{C}}^{0.5}. \blacktriangleleft$$

Returning to the proof of Lemma 15, consider the sequence

$$\begin{aligned}\delta_0 &:= 2^{-n(1-H(p)-\frac{\varepsilon}{1+\varepsilon})}\\ \delta_i &:= 2\delta_{i-1} + \delta_{i-1}^{1.5}.\end{aligned}$$

We prove by induction that, for $0 \le i \le n(1-H(p)-\varepsilon)$, we have $\delta_i < 2^{i+1}\delta_0$, which is at most $2^{-\frac{\varepsilon^2 n}{2}}$ for n sufficiently large. The base case $i = 0$ is straightforward. If $\delta_j < 2^{j+1}\delta_0$ for $j < i$, then

$$\delta_i \;=\; 2\delta_{i-1}(1+\delta_{i-1}^{0.5}) \;=\; 2^i\delta_0 \cdot \prod_{j=0}^{i-1}(1+\delta_j^{0.5}) \;\le\; 2^i\delta_0 \cdot \exp\left(\sum_{j=0}^{i-1}\delta_j^{0.5}\right) \;<\; 2^{i+1}\delta_0.$$

In the first two equalities, we applied the definitions of δ_i and $\delta_{i-1}, \dots, \delta_1$, respectively. In the first inequality, we used the estimate $1 + z \le e^z$, and in the second we used the inductive hypothesis $\delta_j < 2^{-\frac{\varepsilon^2 n}{2}}$ for $j < i$. By this induction, we conclude that, if $k = n(1-H(p)-\varepsilon)$, then $\delta_k < 2^{-\frac{\varepsilon^2 n}{2}}$.

Let $b_1, \dots, b_k \in \mathbb{F}_2^n$ be randomly chosen vectors, and let $\mathcal{C}_i = \operatorname{span}(b_1, \dots, b_i)$ with $\mathcal{C}_k = \mathcal{C}$. By Lemma 16, conditioned on a fixed $\mathcal{C}_i$ satisfying $T_{\mathcal{C}_i} \le \delta_i$, we have, with probability at most $T_{\mathcal{C}_i}^{0.5}$, which is at most $\delta_i^{0.5}$, that $T_{\mathcal{C}_{i+1}} > \delta_{i+1}$. Furthermore, $T_{\mathcal{C}_0} \le \delta_0$ by the initial condition (4). Thus, with probability at least

$$1 - \left(\delta_0^{0.5} + \delta_1^{0.5} + \cdots + \delta_k^{0.5}\right) \;>\; 1 - k2^{-\varepsilon^2 n/2} \;\ge\; 1 - 2^{-\Omega_\varepsilon(n)}$$

we have $T_{\mathcal{C}_i} \le \delta_i$ for all i. In particular, $T_{\mathcal{C}} = T_{\mathcal{C}_k} < \delta_k < 2^{-\frac{\varepsilon^2 n}{2}}$. Thus, $S_{\mathcal{C}} = 1 + T_{\mathcal{C}} \le 2$ with probability $1 - \exp(-\Omega_\varepsilon(n))$, completing the proof of Lemma 15. ◀

▶ Remark. We do not see how to extend this proof to larger alphabets. If, for example, $q = 3$, then Lemma 16 would need to say $\mathbf{Pr}[S_{C+\{0,b,2b\}} > 1 + 3T_C + o(T_C)] < o(1)$. However, the same proof would fail to establish this, as we can no longer separate the expectation in (5); that is we cannot say

$$\mathop{\mathbf{E}}_{b}\mathop{\mathbf{E}}_{x}\left[B_{\mathcal{C}}(x)B_{\mathcal{C}}(x+b)B_{\mathcal{C}}(x+2b)\right] = \mathop{\mathbf{E}}_{b,x}\left[B_{\mathcal{C}}(x)\right] \cdot \mathop{\mathbf{E}}_{b,x}\left[B_{\mathcal{C}}(x+b)\right] \cdot \mathop{\mathbf{E}}_{b,x}\left[B_{\mathcal{C}}(x+2b)\right] = T_{\mathcal{C}}^3.$$

4 Conclusion

In this work, we have given an improved analysis of the list-decodability of random linear binary codes. Our analysis works for all values of p, and also obtains improved bounds on the list size as the rate approaches list-decoding capacity. In particular, not only do our bounds improve on previous work for random linear codes, but they show that random linear codes are more list-decodable than completely random codes, in the sense that the list size is strictly smaller. Our techniques are quite simple, and strengthen an argument of [13] to hold with high probability. In order to demonstrate the applicability of these techniques, we use them to (a) obtain more information about the distribution of list sizes of random linear codes and (b) to prove a similar result for random linear rank-metric codes, improving a recent result of [16].

We end with some open questions raised by our work.

1. With the exception of Theorem 12, our results – both our upper bounds and our lower bounds – hold only for binary alphabets. We conjecture that analogous results, and in particular list-decoding random linear codes with list size C/ε for $C < 1$, hold over larger alphabets.

2. We showed that random linear binary codes of rate $1 - H(p) - \varepsilon$ are with high probability (p, L) list-decodable with $L \leq H(p)/\varepsilon$. The lower bounds of [32, 15] show that we must have $L \geq C/\varepsilon$, but the constant C is much smaller than $H(p)$. Thus, we still do not know what the correct leading constant is for random linear codes.
3. Finally, there are currently no known explicit constructions of capacity-achieving binary list-decodable codes for general p. It is our hope that this work – which gives more information about the structure of linear codes which achieve list-decoding capacity – could lead to progress on this front. Given that we don't know how to efficiently check if a given code is (p, L)-list-decodable, even an efficient Las Vegas construction (as opposed to a Monte Carlo construction) would be interesting.

References

1 Volodia M. Blinovsky. Bounds for codes in the case of list decoding of finite volume. *Problems of Information Transmission*, 22(1):7–19, 1986.

2 Mahdi Cheraghchi, Venkatesan Guruswami, and Ameya Velingker. Restricted isometry of Fourier matrices and list decodability of random linear codes. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 432–442. ACM-SIAM, 2013. doi:10.1137/1.9781611973105.31.

3 Philippe Delsarte. Bilinear forms over a finite field, with applications to coding theory. *Journal of Combinatorial Theory, Series A*, 25(3):226–241, 1978.

4 Yang Ding. On list-decodability of random rank metric codes and subspace codes. *IEEE Trans. Information Theory*, 61(1):51–59, 2015. doi:10.1109/TIT.2014.2371915.

5 Zeev Dvir and Shachar Lovett. Subspace evasive sets. In *Proceedings of the Forty-Fourth annual ACM Symposium on Theory of Computing (STOC)*, pages 351–358. ACM, 2012.

6 Peter Elias. List decoding for noisy channels. *Wescon Convention Record, Part 2*, pages 94–104, 1957.

7 Peter Elias. Error-correcting codes for list decoding. *IEEE Trans. Information Theory*, 37(1):5–12, 1991.

8 Ernst M. Gabidulin, A. V. Paramonov, and O. V. Tretjakov. Ideals over a non-commutative ring and their applications in cryptology. In *Proceedings of Advances in Cryptology - EUROCRYPT '91, Workshop on the Theory and Application of of Cryptographic Techniques*, pages 482–489, 1991. doi:10.1007/3-540-46416-6_41.

9 Ernst Mukhamedovich Gabidulin. Theory of codes with maximum rank distance. *Problemy Peredachi Informatsii*, 21(1):3–16, 1985.

10 Maximilien Gadouleau and Zhiyuan Yan. On the decoder error probability of bounded rank-distance decoders for maximum rank-distance codes. *IEEE Trans. Information Theory*, 54(7):3202–3206, 2008. doi:10.1109/TIT.2008.924697.

11 Venkatesan Guruswami. List decoding of binary codes-a brief survey of some recent results. In *Proceedings of Coding and Cryptology, Second International Workshop (IWCC)*, pages 97–106, 2009. doi:10.1007/978-3-642-01877-0_10.

12 Venkatesan Guruswami, Johan Håstad, and Swastik Kopparty. On the list-decodability of random linear codes. *IEEE Trans. Information Theory*, 57(2):718–725, 2011. doi:10.1109/TIT.2010.2095170.

13 Venkatesan Guruswami, Johan Håstad, Madhu Sudan, and David Zuckerman. Combinatorial bounds for list decoding. *IEEE Trans. Information Theory*, 48(5):1021–1034, 2002. doi:10.1109/18.995539.

14 Venkatesan Guruswami and Piotr Indyk. Efficiently decodable codes meeting gilbert-varshamov bound for low rates. In *Proceedings of the fifteenth annual ACM-SIAM sym-*

posium on Discrete algorithms, pages 756–757. Society for Industrial and Applied Mathematics, 2004.

15 Venkatesan Guruswami and Srivatsan Narayanan. Combinatorial limitations of average-radius list-decoding. *IEEE Trans. Information Theory*, 60(10):5827–5842, 2014. doi:10.1109/TIT.2014.2343224.

16 Venkatesan Guruswami and Nicolas Resch. On the list-decodability of random linear rank-metric codes. *arXiv preprint arXiv:1710.11516*, 2017.

17 Venkatesan Guruswami and Atri Rudra. Concatenated codes can achieve list-decoding capacity. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2008, San Francisco, California, USA, January 20-22, 2008*, pages 258–267, 2008. URL: http://dl.acm.org/citation.cfm?id=1347082.1347111.

18 Venkatesan Guruswami and Atri Rudra. Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy. *IEEE Trans. Information Theory*, 54(1):135–150, 2008.

19 Venkatesan Guruswami and Salil Vadhan. A lower bound on list size for list decoding. In Chandra Chekuri, Klaus Jansen, José D. P. Rolim, and Luca Trevisan, editors, *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 318–329, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

20 Venkatesan Guruswami, Carol Wang, and Chaoping Xing. Explicit list-decodable rank-metric and subspace codes via subspace designs. *IEEE Trans. Information Theory*, 62(5):2707–2718, 2016.

21 Venkatesan Guruswami and Chaoping Xing. Folded codes from function field towers and improved optimal rate list decoding. In *Proceedings of the Forty-Fourth annual ACM Symposium on Theory of Computing (STOC)*, pages 339–350. ACM, 2012.

22 Venkatesan Guruswami and Chaoping Xing. List decoding Reed-Solomon, Algebraic-Geometric, and Gabidulin subcodes up to the Singleton bound. In *Proceedings of the Forty-Fifth annual ACM Symposium on Theory of Computing (STOC)*, pages 843–852. ACM, 2013.

23 Brett Hemenway, Noga Ron-Zewi, and Mary Wootters. Local list recovery of high-rate tensor codes & applications. In *58th Annual IEEE Symposium on Foundations of Computer Science*, 2017.

24 Brett Hemenway and Mary Wootters. Linear-time list recovery of high-rate expander codes. In *International Colloquium on Automata, Languages, and Programming*, pages 701–712. Springer, 2015.

25 Ralf Koetter and Frank R Kschischang. Coding for errors and erasures in random network coding. *IEEE Trans. Information Theory*, 54(8):3579–3591, 2008.

26 Ray Li and Mary Wootters. Improve list-decodability of random linear binary code. *CoRR*, abs/1801.07839, 2018. arXiv:1801.07839.

27 Pierre Loidreau. Designing a rank metric based mceliece cryptosystem. In *Proceedings of the Post-Quantum Cryptography, Third International Workshop on Post-Quantum Cryptography, (PQCrypto)*, pages 142–152, 2010. doi:10.1007/978-3-642-12929-2_11.

28 Pierre Loidreau. A new rank metric codes based encryption scheme. In *Proceedings of the Post-Quantum Cryptography, 8th International Workshop on Post-Quantum Cryptography, (PQCrypto)*, pages 3–17, 2017. doi:10.1007/978-3-319-59879-6_1.

29 Hsiao-feng Lu and P. Vijay Kumar. A unified construction of space-time codes with optimal rate-diversity tradeoff. *IEEE Trans. Information Theory*, 51(5):1709–1730, 2005. doi:10.1109/TIT.2005.846403.

30 P. Lusina, Ernst M. Gabidulin, and Martin Bossert. Maximum rank distance codes as space-time codes. *IEEE Trans. Information Theory*, 49(10):2757–2760, 2003. doi:10.1109/TIT.2003.818023.

31 Ron M. Roth. Maximum-rank array codes and their application to crisscross error correction. *IEEE Trans. Information Theory*, 37(2):328–336, 1991. doi:10.1109/18.75248.

32 Atri Rudra. Limits to list decoding of random codes. *IEEE Trans. Information Theory*, 57(3):1398–1408, 2011. doi:10.1109/TIT.2010.2054750.

33 Atri Rudra and Steve Uurtamo. Two theorems on list decoding. In Maria Serna, Ronen Shaltiel, Klaus Jansen, and José Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 696–709, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

34 Atri Rudra and Mary Wootters. Every list-decodable code for high noise has abundant near-optimal rate puncturings. In *Proceedings of the Forty-Sixth annual ACM Symposium on Theory of Computing (STOC)*, pages 764–773. ACM, 2014.

35 Atri Rudra and Mary Wootters. It'll probably work out: Improved list-decoding through random operations. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science (ITCS)*, pages 287–296. ACM, 2015. doi:10.1145/2688073.2688092.

36 Atri Rudra and Mary Wootters. Average-radius list-recovery of random linear codes. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. ACM-SIAM, 2018.

37 Natalia Silberstein, Ankit Singh Rawat, O Ozan Koyluoglu, and Sriram Vishwanath. Optimal locally repairable codes via rank-metric codes. In *Proceedings of the 2013 IEEE International Symposium on Information Theory (ISIT)*, pages 1819–1823. IEEE, 2013.

38 Natalia Silberstein, Ankit Singh Rawat, and Sriram Vishwanath. Error resilience in distributed storage via rank-metric codes. In *Proceedings of the 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1150–1157. IEEE, 2012.

39 Danilo Silva, Frank R Kschischang, and Ralf Koetter. A rank-metric approach to error control in random network coding. *IEEE Trans. Information Theory*, 54(9):3951–3967, 2008.

40 Madhu Sudan. List decoding: algorithms and applications. *SIGACT News*, 31(1):16–27, 2000. doi:10.1145/346048.346049.

41 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1-3):1–336, 2012. doi:10.1561/0400000010.

42 Antonia Wachter-Zeh. Bounds on list decoding of rank-metric codes. *IEEE Trans. Information Theory*, 59(11):7268–7277, 2013.

43 Mary Wootters. On the list decodability of random linear codes with large error rates. In *Proceedings of the Forty-Fifth Symposium on Theory of Computing Conference (STOC)*, pages 853–860, 2013. doi:10.1145/2488608.2488716.

44 Jack Wozencraft. List decoding. *Quarter Progress Report*, 48:90–95, 1958.

45 Victor Vasilievich Zyablov and Mark Semenovich Pinsker. List concatenated decoding. *Problemy Peredachi Informatsii*, 17(4):29–33, 1981.

A Characterizing the list size distribution

In this section we establish Theorem 6. Define

$$\begin{aligned} P_{\mathcal{C}}^{(\ell)} &:= 2^{-n}|\{x : L_{\mathcal{C}}(x) = \ell\}| \\ P_{\mathcal{C}}^{(\geq \ell)} &:= \sum_{i=\ell}^{\infty} P_{\mathcal{C}}^{(i)} \\ Q_{\mathcal{C}}^{(\geq \ell)} &:= \sum_{i=\ell}^{\infty} i \cdot P_{\mathcal{C}}^{(i)} \end{aligned}$$

The goal (per Theorem 6) is to bound $P_{\mathcal{C}}^{(\geq \ell)}$; in our argument, it is more convenient to work with $Q_{\mathcal{C}}^{(\geq \ell)}$, which is a proxy for $P_{\mathcal{C}}^{(\geq \ell)}$. We note a few useful properties of these definitions.

▶ **Proposition 17.** *Suppose that $\mathcal{C} \subseteq \mathbb{F}_2^n$ is a linear code. Then the following hold for $\ell \geq 1$.*
1. $Q_{\mathcal{C}}^{(\geq \ell)} = 2^{-n} \cdot \sum_{x \in \mathbb{F}_2^n : L_{\mathcal{C}}(x) \geq \ell} L_{\mathcal{C}}(x) = \mathbf{E}_x \left[\mathbb{I}\left[L_{\mathcal{C}}(x) \geq \ell \right] \cdot L_{\mathcal{C}}(x) \right]$
2. $Q_{\mathcal{C}}^{(\geq 1)} = 2^{-n} \cdot \mathrm{Vol}(n, pn) \cdot 2^k \leq 2^{-n(1-H(p))+k}$

Proof. To see Item 1, notice that

$$\begin{aligned} Q_{\mathcal{C}}^{(\geq \ell)} &= 2^{-n} \sum_{i=\ell}^{\infty} i \cdot |\{x \mid L_{\mathcal{C}}(x) = i\}| \\ &= 2^{-n} \sum_{i=\ell}^{\infty} \sum_{x \in \mathbb{F}_2^n} i \cdot \mathbb{I}\left[L_{\mathcal{C}}(x) = i \right] \\ &= 2^{-n} \sum_{x : L_{\mathcal{C}}(x) \geq \ell} L_{\mathcal{C}}(x). \end{aligned}$$

To see Item 2, we begin with Item 1 and derive

$$\begin{aligned} Q_{\mathcal{C}}^{(\geq 1)} &= 2^{-n} \sum_{x : L_{\mathcal{C}}(x) \geq 1} L_{\mathcal{C}}(x) \\ &= 2^{-n} \sum_{x \in \mathbb{F}_2^n} L_{\mathcal{C}}(x) \\ &= 2^{-n} \sum_{x \in \mathbb{F}_2^n} \sum_{v \in \mathcal{B}(x, pn)} \mathbb{I}[x + v \in \mathcal{C}] \\ &= 2^{-n} \sum_{v \in \mathcal{B}(x, pn)} \sum_{x \in \mathbb{F}_2^n} \mathbb{I}[x + v \in \mathcal{C}] \\ &= 2^{-n} \sum_{v \in \mathcal{B}(x, pn)} |\mathcal{C}| \\ &= 2^{-n} \cdot \mathrm{Vol}(n, pn) \cdot 2^k. \end{aligned}$$

◀

Using these properties, we are now can state and prove Theorem 18, which implies Theorem 6.

▶ **Theorem 18.** *Fix $L \geq 0$. There exists a constant C_L depending only on L so that, for all $\gamma \in (0,1)$ and sufficiently large n, the following holds. Suppose that $k \leq (1-\gamma)n$. If $\mathcal{C} \subseteq \mathbb{F}_2^n$ is a random linear code of dimension k, then with probability $1 - \exp(-C_L \gamma n)$, for all $1 \leq \ell \leq L$,*

$$Q_{\mathcal{C}}^{(\geq \ell)} \leq \left(2^{-n(1-H(p))} \cdot 2^k \right)^{\ell} \cdot 2^{\gamma \ell^2 n}. \tag{6}$$

Before we prove Theorem 18, we explain why it implies Theorem 6. By setting $k = n(1 - H(p) - \varepsilon)$ and $\gamma = \varepsilon\gamma'/L$ in Theorem 18, we obtain that with high probability, for all $1 \leq \ell \leq L$,

$$P_{\mathcal{C}}^{(\geq \ell)} \leq Q_{\mathcal{C}}^{(\geq \ell)} \leq 2^{-n\varepsilon\ell} \cdot 2^{\frac{\varepsilon\gamma'}{L}\ell^2 n} \leq 2^{-n\varepsilon\ell(1-\gamma')},$$

which is Theorem 6. Theorem 18 also implies Theorem 5:

▶ **Corollary 19** (Theorem 5). *Let $p \in (0, 1/2)$, let $\varepsilon > 0$, and let $R = 1 - H(p) - \varepsilon$. Let $\mathcal{C} \subseteq \mathbb{F}_2^n$ be a random linear code of rate R. Then with probability $1 - \exp(-\Omega_\varepsilon(n))$, the code $\mathcal{C}$ is $(p, H(p)/\varepsilon + 2)$-list-decodable.*

Proof. Let $L = H(p)/\varepsilon + 2$ and choose $\gamma \ll L^{-3}$, and let $\mathcal{C} \subseteq \mathbb{F}_2^n$ be a random linear code of dimension $k = n(1 - H(p) - \varepsilon)$. Then $k \leq (1-\gamma)n$, so, by Theorem 18, with probability at least $1 - \exp(-C_L \gamma n)$, code $\mathcal{C}$ satisfies (6) with $k = n(1 - H(p) - \varepsilon)$. In particular, choosing $\ell = L$, this means that

$$Q_{\mathcal{C}}^{(\geq L)} \;\leq\; 2^{L(-n(1-H(p))+k)+\gamma L^2 n} \;=\; 2^{-nL\varepsilon+\gamma L^2 n} \;<\; 2^{-n(1-R)}$$

Suppose there exists $x \in \mathbb{F}_2^n$ such that $L_{\mathcal{C}}(x) \geq L$. Then $L_{\mathcal{C}}(x+c) \geq L$ for all $c \in C$, so that $Q_{\mathcal{C}}^{(\geq L)} > P_{\mathcal{C}}^{(\ell)} \geq 2^{-n(1-R)}$. This is a contradiction, so there are no $x \in \mathbb{F}_2^n$ such that $L_{\mathcal{C}}(x) \geq L$. ◀

Proof of Theorem 18. The proof of Theorem 18 proceeds by induction on k, with the inductive hypothesis that (6) holds for all $1 \leq \ell \leq L$. We begin with the base case by noting that (6) is satisfied for $k = 0$, for all $1 \leq \ell \leq L$. To see this, notice that for $\ell = 1$, we have $Q_{\{0\}}^{(\geq 1)} \leq 2^{-n(1-H(p))}$ by Proposition 17, Item 2, which satisfies (6) for $k = 0, \ell = 1$. For $\ell \geq 2$, $Q_{\{0\}}^{(\geq \ell)} = 0$, and so again (6) holds.

Now that we have established the base case of $k = 0$, we proceed by induction. Lemma 20 provides the inductive step; similar to the approach in §3, it shows that at every step the "expected behavior" holds with high probability.

▶ **Lemma 20.** *Let $\gamma > 0$, and suppose that $\mathcal{C} \subseteq \mathbb{F}_2^n$ is a linear code of dimension $k \leq (1-\gamma)n$ such that for all $1 \leq \ell \leq L$, we have*

$$Q_{\mathcal{C}}^{(\geq \ell)} \;\leq\; \left(2^{-n(1-H(p))} \cdot 2^k\right)^{\ell} \cdot 2^{\gamma \ell^2 n}. \tag{7}$$

Then, for a uniformly chosen $b \in \mathbb{F}_2^n$, with probability at least $1 - 4 \cdot L^3 \cdot 2^{-\gamma n}$ over the choice of b, we have, for all $1 \leq \ell \leq L$,

$$Q_{\mathcal{C}+\{0,b\}}^{(\geq \ell)} \;\leq\; \left(2^{-n(1-H(p))} \cdot 2^{k+1}\right)^{\ell} \cdot 2^{\gamma \ell^2 n}. \tag{8}$$

Proof. By Proposition 17, Item 2, (8) always holds for $\ell = 1$, so suppose that $\ell \geq 2$. We have, for any $b \in \mathbb{F}_2^n$,

$$\begin{aligned}
Q_{\mathcal{C}+\{0,b\}}^{(\geq \ell)} &= \mathop{\mathbf{E}}_x \left[\mathbb{I}[L_{\mathcal{C}+\{0,b\}}(x) \geq \ell] \cdot L_{\mathcal{C}+\{0,b\}}(x)\right] && \text{(By Prop. 17, Item 1)} \\
&\leq \mathop{\mathbf{E}}_x \left[\mathbb{I}[L_{\mathcal{C}}(x) + L_{\mathcal{C}}(x+b) \geq \ell] \cdot (L_{\mathcal{C}}(x) + L_{\mathcal{C}}(x+b))\right] && \text{(By Lemma 14)} \\
&= \mathop{\mathbf{E}}_x \left[\mathbb{I}[L_{\mathcal{C}}(x) + L_{\mathcal{C}}(x+b) \geq \ell] \cdot L_{\mathcal{C}}(x) + \mathbb{I}[L_{\mathcal{C}}(x) + L_{\mathcal{C}}(x+b) \geq \ell] \cdot L_{\mathcal{C}}(x+b)\right] \\
&= 2 \cdot \mathop{\mathbf{E}}_x \left[\mathbb{I}[L_{\mathcal{C}}(x) + L_{\mathcal{C}}(x+b) \geq \ell] \cdot L_{\mathcal{C}}(x)\right] \\
&= 2 \cdot \mathop{\mathbf{E}}_x \left[\mathbb{I}[L_{\mathcal{C}}(x) \geq \ell] \cdot L_{\mathcal{C}}(x) + \sum_{i=0}^{\ell-1} \mathbb{I}[L_{\mathcal{C}}(x) = i, L_{\mathcal{C}}(x+b) \geq \ell - i] \cdot L_{\mathcal{C}}(x)\right] \\
&= 2 \cdot Q_{\mathcal{C}}^{(\geq \ell)} + 2 \cdot \sum_{i=0}^{\ell-1} i \cdot \mathop{\mathbf{E}}_x \left[\mathbb{I}[L_{\mathcal{C}}(x) = i] \cdot \mathbb{I}[L_{\mathcal{C}}(x+b) \geq \ell - i]\right], && (9)
\end{aligned}$$

where we have used Proposition 17, Item 1 in the final line. Since the only inequality above is an application of Lemma 14, which is an equality when $b \notin \mathcal{C}$, the derivation above is an equality when $b \notin \mathcal{C}$. Thus, when $b \notin \mathcal{C}$, we have

$$Q_{\mathcal{C}+\{0,b\}}^{(\geq \ell)} = 2 \cdot Q_{\mathcal{C}}^{(\geq \ell)} + 2 \cdot \sum_{i=0}^{\ell-1} i \cdot \mathop{\mathbf{E}}_x \left[\mathbb{I}[L_{\mathcal{C}}(x) = i] \cdot \mathbb{I}[L_{\mathcal{C}}(x+b) \geq \ell - i]\right] \geq 2 \cdot Q_{\mathcal{C}}^{(\geq \ell)}. \tag{10}$$

This fact (10) is useful later. For now, we move on with no assumption on b. Taking expectations on both sides of (9), we see

$$\begin{aligned}
\mathop{\mathbf{E}}_{b}\left[Q^{(\geq \ell)}_{\mathcal{C}+\{0,b\}}\right] &\leq 2\cdot Q^{(\geq \ell)}_{\mathcal{C}} + 2\cdot \sum_{i=0}^{\ell-1} i \cdot \mathop{\mathbf{E}}_{b,x}\left[\mathbb{I}[L_{\mathcal{C}}(x)=i]\cdot \mathbb{I}[L_{\mathcal{C}}(x+b)\geq \ell - i]\right] \\
&= 2\cdot Q^{(\geq \ell)}_{\mathcal{C}} + 2\cdot \sum_{i=0}^{\ell-1} i \cdot P^{(i)}_{\mathcal{C}}\cdot P^{(\geq \ell-i)}_{\mathcal{C}} \\
&\leq 2\cdot Q^{(\geq \ell)}_{\mathcal{C}} + 2\cdot \sum_{i=0}^{\ell-1} i \cdot Q^{(\geq i)}_{\mathcal{C}}\cdot Q^{(\geq \ell-i)}_{\mathcal{C}}
\end{aligned}$$

Continuing, we bound

$$\begin{aligned}
\mathop{\mathbf{E}}_{b}\left[Q^{(\geq \ell)}_{\mathcal{C}+\{0,b\}}\right] &\leq 2\cdot Q^{(\geq \ell)}_{\mathcal{C}} + 2\cdot \sum_{i=0}^{\ell-1} i \cdot Q^{(\geq i)}_{\mathcal{C}}\cdot Q^{(\geq \ell-i)}_{\mathcal{C}} \\
&\leq 2\cdot Q^{(\geq \ell)}_{\mathcal{C}} + 2\cdot L\cdot \sum_{i=1}^{\ell-1} Q^{(\geq i)}_{\mathcal{C}}\cdot Q^{(\geq \ell-i)}_{\mathcal{C}} \\
&\leq 2\cdot Q^{(\geq \ell)}_{\mathcal{C}} + 2\cdot L\cdot \sum_{i=1}^{\ell-1} \left(2^{-n(1-H(p))}\cdot 2^{k}\right)^{\ell}\cdot 2^{\gamma(\ell^2-2i\ell+2i^2)n} \qquad \text{(By (7))} \\
&\leq 2\cdot Q^{(\geq \ell)}_{\mathcal{C}} + 2\cdot L\cdot \sum_{i=1}^{\ell-1} \left(2^{-n(1-H(p))}\cdot 2^{k}\right)^{\ell}\cdot 2^{\gamma\ell^2 n}\cdot 2^{-\gamma n} \\
&\leq 2\cdot Q^{(\geq \ell)}_{\mathcal{C}} + 2\cdot L^2\cdot 2^{-\gamma n}\cdot \left(2^{-n(1-H(p))}\cdot 2^{k}\right)^{\ell}\cdot 2^{\gamma\ell^2 n} \\
&=: (\star)
\end{aligned}$$

The above derivation holds whether or not $b\in\mathcal{C}$. Thus,

$$\begin{aligned}
(\star) \geq \mathop{\mathbf{E}}_{b}\left[Q^{(\geq \ell)}_{\mathcal{C}+\{0,b\}}\right] &= \mathop{\mathbf{E}}_{b}\left[Q^{(\geq \ell)}_{\mathcal{C}+\{0,b\}} | b\notin\mathcal{C}\right]\mathop{\mathbf{Pr}}_{b}\left[b\notin\mathcal{C}\right] + \mathop{\mathbf{E}}_{b}\left[Q^{(\geq \ell)}_{\mathcal{C}+\{0,b\}} | b\in\mathcal{C}\right]\mathop{\mathbf{Pr}}_{b}\left[b\in\mathcal{C}\right] \\
&\geq \mathop{\mathbf{E}}_{b}\left[Q^{(\geq \ell)}_{\mathcal{C}+\{0,b\}} | b\notin\mathcal{C}\right]\mathop{\mathbf{Pr}}_{b}\left[b\notin\mathcal{C}\right] \\
&= \mathop{\mathbf{E}}_{b}\left[Q^{(\geq \ell)}_{\mathcal{C}+\{0,b\}} | b\notin\mathcal{C}\right]\cdot\left(1-\frac{|\mathcal{C}|}{2^n}\right),
\end{aligned}$$

and so

$$\begin{aligned}
&\mathop{\mathbf{E}}_{b}\left[Q^{(\geq \ell)}_{\mathcal{C}+\{0,b\}} | b\notin\mathcal{C}\right] \\
&\quad\leq \frac{2\cdot Q^{(\geq \ell)}_{\mathcal{C}} + 2\cdot L^2\cdot 2^{-\gamma n}\cdot\left(2^{-n(1-H(p))}\cdot 2^{k}\right)^{\ell}\cdot 2^{\gamma\ell^2 n}}{1-\frac{|C|}{2^n}} \\
&\quad\leq \left(1+\frac{2|C|}{2^n}\right)\left(2\cdot Q^{(\geq \ell)}_{\mathcal{C}} + 2\cdot L^2\cdot 2^{-\gamma n}\cdot\left(2^{-n(1-H(p))}\cdot 2^{k}\right)^{\ell}\cdot 2^{\gamma\ell^2 n}\right) \\
&\quad\leq 2\cdot Q^{(\geq \ell)}_{\mathcal{C}} + \left(2^{-n(1-H(p))}\cdot 2^{k}\right)^{\ell}\cdot 2^{\gamma\ell^2 n}\left(\frac{4|C|}{2^n} + 3\cdot L^2\cdot 2^{-\gamma n}\right) \qquad \text{(By (7))} \\
&\quad\leq 2\cdot Q^{(\geq \ell)}_{\mathcal{C}} + 4\cdot L^2\cdot 2^{-\gamma n}\cdot\left(2^{-n(1-H(p))}\cdot 2^{k}\right)^{\ell}\cdot 2^{\gamma\ell^2 n}.
\end{aligned}$$

In the third line, we used the fact that $1/(1-x)\leq 1+2x$ for all $0\leq x\leq 1/2$, along with the fact that, since $k<n$, we have $|\mathcal{C}|/2^n = 2^{k-n}\leq 1/2$. In the last line, we used that

$|\mathcal{C}| = 2^k \leq 2^{n(1-\gamma)}$ and $2 \leq \ell \leq L$. By (10), when $b \notin \mathcal{C}$, the quantity $Q^{(\geq \ell)}_{\mathcal{C}+\{0,b\}} - 2Q^{(\geq \ell)}_{\mathcal{C}}$ is nonnegative. Hence, we may apply Markov's inequality to obtain

$$\mathop{\mathbf{Pr}}_{b}\left[Q^{(\geq \ell)}_{\mathcal{C}+\{0,b\}} - 2Q^{(\geq \ell)}_{\mathcal{C}} \geq \left(2^{-n(1-H(p))} \cdot 2^k\right)^{\ell} \cdot 2^{\gamma \ell^2 n} | b \notin \mathcal{C}\right] \;\leq\; 4 \cdot L^2 \cdot 2^{-\gamma n}.$$

When $b \in \mathcal{C}$, we have $Q^{(\geq \ell)}_{\mathcal{C}+\{0,b\}} = Q^{(\geq \ell)}_{\mathcal{C}}$. Thus,

$$\mathop{\mathbf{Pr}}_{b}\left[Q^{(\geq \ell)}_{\mathcal{C}+\{0,b\}} - 2Q^{(\geq \ell)}_{\mathcal{C}} \geq \left(2^{-n(1-H(p))} \cdot 2^k\right)^{\ell} \cdot 2^{\gamma \ell^2 n} | b \in \mathcal{C}\right] = 0.$$

Together these imply

$$\mathop{\mathbf{Pr}}_{b}\left[Q^{(\geq \ell)}_{\mathcal{C}+\{0,b\}} - 2Q^{(\geq \ell)}_{\mathcal{C}} \geq \left(2^{-n(1-H(p))} \cdot 2^k\right)^{\ell} \cdot 2^{\gamma \ell^2 n}\right] \;\leq\; 4 \cdot L^2 \cdot 2^{-\gamma n}. \tag{11}$$

Thus, with probability at least $1 - 4L^2 2^{-\gamma n}$, we have

$$\begin{aligned} Q^{(\geq \ell)}_{\mathcal{C}+\{0,b\}} &\leq 2Q^{(\geq \ell)}_{\mathcal{C}} + \left(2^{-n(1-H(p))} \cdot 2^k\right)^{\ell} \cdot 2^{\gamma \ell^2 n} \\ &\leq 3\left(2^{-n(1-H(p))} \cdot 2^k\right)^{\ell} \cdot 2^{\gamma \ell^2 n} \\ &\leq \left(2^{-n(1-H(p))} \cdot 2^{k+1}\right)^{\ell} \cdot 2^{\gamma \ell^2 n} \end{aligned}$$

where the first inequality is from (11), the second inequality is by the assumption (7), and the final inequality is because $\ell \geq 2$. This completes the proof of Lemma 20. ◀

Returning to the proof of Theorem 18, call a code $\mathcal{C}$ of dimension k *good* if (6) holds for all $1 \leq \ell \leq L$. Lemma 20 states that if $\mathcal{C}$ is good, then $\mathcal{C} + \{0, b\}$ fails to be good with probability at most $4L^3 2^{-\gamma n}$ over the choice of a uniformly random $b \in \mathbb{F}_2^n$.

Since we have already shown that $\{0\}$ is good at the beginning of the proof, it follows from the union bound that a random linear binary code $\mathcal{C} = \mathrm{span}(b_1, \ldots, b_k)$ fails to be good with probability at most $k \cdot 4L^3 2^{-\gamma n} = 2^{-\Omega(\gamma n)}$. This completes the proof of Theorem 18. ◀

Sunflowers and Quasi-Sunflowers from Randomness Extractors

Xin Li[1]
Johns Hopkins University, Baltimore, USA
lixints@cs.jhu.edu

Shachar Lovett[2]
University of California San Diego, La Jolla, USA
slovett@cs.ucsd.edu

Jiapeng Zhang[3]
University of California San Diego, La Jolla, USA
jpeng.zhang@gmail.com

Abstract

The Erdős-Rado sunflower theorem (Journal of Lond. Math. Soc. 1960) is a fundamental result in combinatorics, and the corresponding sunflower conjecture is a central open problem. Motivated by applications in complexity theory, Rossman (FOCS 2010) extended the result to quasi-sunflowers, where similar conjectures emerge about the optimal parameters for which it holds.

In this work, we exhibit a surprising connection between the existence of sunflowers and quasi-sunflowers in large enough set systems, and the problem of constructing (or existing) certain randomness extractors. This allows us to re-derive the known results in a systematic manner, and to reduce the relevant conjectures to the problem of obtaining improved constructions of the randomness extractors.

2012 ACM Subject Classification Theory of computation → Randomness, geometry and discrete structures

Keywords and phrases Sunflower conjecture, Quasi-sunflowers, Randomness Extractors

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.51

Acknowledgements We would like to thank Raghu Meka for helpful discussions. We thank anonymous reviewers for insightful suggestions.

1 Introduction

Let $\mathcal{F}$ be a collection of sets from some universe X. A common theme and extensively studied phenomenon in combinatorics is the following: if the cardinality of $\mathcal{F}$ (when $\mathcal{F}$ is finite) or the density of $\mathcal{F}$ (when $\mathcal{F}$ is infinite) is large enough, then some nice patterns will occur in $\mathcal{F}$. Well known examples of this kind include (1) Szemerédi's theorem [20], which asserts that all subsets of the natural numbers of positive density contain arbitrarily long arithmetic progressions; (2) Ramsey's theorem [10], which asserts that if one colors the edges of a large enough complete graph with a finite number of colors, then there must exist a monochromatic clique of a certain size; and (3) the Erdős-Rado sunflower theorem [9], which

[1] Research supported by NSF award CCF-1617713
[2] Research supported by NSF CCF-1614023
[3] Research supported by NSF CCF-1614023

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018).
Editors: Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer; Article No. 51; pp. 51:1–51:13

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

asserts that a large enough collection of subsets with bounded size of a universe must contain a large sunflower.[4]

The study of these problems has resulted in many important tools (e.g., Szemerédi's regularity lemma [21] and the probabilistic method), which have found wide applications not only in combinatorics, but also in computer science. Conversely, ideas from computer science have also influenced related research in combinatorics quite often. For example, the first two problems we mentioned above, Szemerédi's theorem and Ramsey's theorem, are intimately connected to the area of pseudorandomness in theoretical computer science. Indeed, by constructing a certain sparse pseudorandom subset of natural numbers and proving an appropriate Szemerédi-type theorem with respect to that subset, a celebrated result of Green and Tao [13] shows that prime numbers contain arbitrarily long arithmetic progressions. As for Ramsey's theorem, a recent line of work on randomness extractors [2], [15], [4], [3], [6], [16] give highly explicit constructions of Ramsey graphs that almost match the probabilistic bound [8].

In this paper we study the sunflower theorem and its related variants. We show that again there is an intimate connection to randomness extractors. In fact, using the techniques from randomness extractors, we build a general proof framework that can unify the sunflower theorem and its variant known as the quasi-sunflower lemma [18]. Furthermore, any improvement in the analysis of the extractors will lead to improvements in the lemmas. We now begin our formal discussion of the sunflower theorem and the quasi-sunflower lemma.

Sunflowers

An *r-sunflower* is defined to be a collection of r sets from some universe X, such that the intersection of any two sets is the same (which can be the empty set). Choose any collection $\mathcal{F}$ of sets from X, the main question of interest is how large $\mathcal{F}$ needs to be in order to ensure that there is a r-sunflower in $\mathcal{F}$. Erdős and Rado proved the following theorem.

▶ **Theorem 1** ([9]). *Let $\mathcal{F}$ be an arbitrary family of sets from some universe X, where each set in $\mathcal{F}$ has size w. If $|\mathcal{F}| > w!(r-1)^w$ then $\mathcal{F}$ contains an r-sunflower.*

They also conjectured that the bound on $|\mathcal{F}|$ can be replaced by c_r^w where c_r is a constant that only depends on r for every $r > 0$. This conjecture is one of the most well known open problems in combinatorics, which remains open today despite a lot of research.

The sunflower theorem has applications in computer science, such as proving strong lower bounds for monotone circuits [17]. In addition, a paper by Alon, Shpilka and Umans [1] relates the sunflower conjecture and its variants to possible approaches of achieving fast matrix multiplication. Recently, following breakthrough results that prove the strong Cap Set Conjecture [7], [19], a weaker version of the sunflower conjecture known as the Erdős-Szemeredi Sunflower Conjecture is also proved. However, the general conjecture still remains open.

Quasi-sunflowers

Motivated by the applications of sunflowers in proving monotone circuit lower bounds, quasi-sunflowers are introduced by Rossman [18] to prove monotone circuit lower bounds for the k-clique problem on random graphs. We denote by $\mathcal{P}(X)$ the family of all subsets of a finite set X.

[4] A sunflower is a collection of sets whose pairwise intersection is constant, which we will formally define shortly.

▶ **Definition 2** ([18]). Let X be a finite set, $\mathcal{S} \subseteq \mathcal{P}(X)$ be a family of sets such that $|\mathcal{S}| \geq 2$. Denote $Y = \bigcap_{T \in \mathcal{S}} T$. For $p, \gamma \in [0,1]$, $\mathcal{S}$ is said to be a (p,γ)-quasi-sunflower if for a random set $W \subseteq X$, with each element of X present in W independently with probability p,

$$\Pr\left[\exists T \in \mathcal{S}, (T \setminus Y) \subseteq W\right] \geq 1 - \gamma.$$

In the same paper, Rossman also proved the following quasi-sunflower lemma, which says there is always a quasi-sunflower in a large family of subsets.

▶ **Lemma 3** (Quasi-sunflower lemma, [18]). *Let $\mathcal{F}$ be a family of sets over a universe X each of size w. If $|\mathcal{F}| \geq w! \cdot (1.71 \log(1/\gamma)/p)^w$, then $\mathcal{F}$ contains a (p,γ)-quasi-sunflower.*

Besides the original application, Rossman's quasi-sunflower lemma was also used by Gopalan, Meka and Reingold [12] to study the problem of DNF sparsification. Given a DNF formula f on n variables, there are two natural ways to measure the complexity of f: the number of clauses (also called size) $s(f)$, and the maximum width of a clause $w(f)$. It is easy to show that any DNF of small size can be approximated well by another DNF of small width, by truncating clauses of larger width. Gopalan et al. [12] used Rossman's quasi-sunflower lemma to show the reverse direction, that any DNF with small width can also be approximated well by another DNF with small size. In particular, they showed that any width w DNF formula can be ε-approximated by another DNF formula with size at most $(w \log(1/\varepsilon))^{O(w)}$. This kind of sparsification has applications in constructing pseudorandom generators and approximately counting the number of satisfying assignments for DNF formulas.

Similar to the sunflower conjecture, one can also ask whether the bound on $\mathcal{F}$ in the quasi-sunflower lemma can be improved. For example, if one can improve the bound to $(O(\log(1/\gamma)/p))^w$ then it is also possible to improve the ε-approximation of DNF formula in [12] to have size $(\log(1/\varepsilon))^{O(w)}$.

1.1 Our contribution

We provide a general framework to prove both the sunflower theorem and the quasi-sunflower lemma. In fact, we reduce both of these problems to the construction of a certain type of randomness extractors. To state our results, we first formally define the notions that are going to be used in our extractors.

▶ **Definition 4.** Let D be a distribution over a sample space X. The min-entropy of D is defined as

$$\mathcal{H}_\infty(D) = \min_x \left\{ \log \left(\frac{1}{\Pr[D = x]} \right) \right\}.$$

▶ **Definition 5** (Block min-entropy source). A distribution $X = (X_1, \cdots, X_m)$ where each $X_i \in \{0,1\}^n$ is an (m, n, k) *block min-entropy* source if for every non-empty subset $S \subseteq [m]$, the joint distribution of $(X_i : i \in S)$ has min-entropy at least $k|S|$.

We note that the definition of block min-entropy sources was initiated in [11] as a tool to prove lifting theorems in communication complexity.

▶ **Definition 6** (Block min-entropy extractor). A function $E : \{0,1\}^n \times \{0,1\}^s \to \{0,1\}^d$ is a (k, ε, d, s) *block min-entropy* extractor if for any $m, n \in \mathsf{N}$ and any (m, n, k) block min-entropy source $X = (X_1, \cdots, X_m)$, we have that

$$(E(X_1, R_1), \cdots, E(X_m, R_m)) \approx_\varepsilon U_{dm}.$$

Here, each $R_i \in \{0,1\}^s$ is an independent uniform random string, U_{dm} is the uniform distribution on dm bits, and $\approx_\varepsilon$ means ε close in the statistical distance. If in addition we have that

$$(E(X_1, R_1), R_1, \cdots, E(X_m, R_m), R_m) \approx_\varepsilon U_{(d+s)m},$$

then we say that the function E is a *strong* (k, ε, d, s) block min-entropy extractor.

We also define a weaker object called a disperser.

▶ **Definition 7** (Block min-entropy disperser)**.** A function

$$E : \{0,1\}^n \times \{0,1\}^s \to \{0,1\}^d$$

is a (k, ε, d, s) *block min-entropy disperser* if for any $m, n \in \mathsf{N}$ and any (m, n, k) block min-entropy source $X = (X_1, \cdots, X_m)$, we have that

$$|\mathsf{Supp}(E(X_1, R_1), \cdots, E(X_m, R_m))| \geq (1-\varepsilon)2^{dm}.$$

Here, each $R_i \in \{0,1\}^s$ is an independent uniform random string, and Supp means the support of the distribution. If in addition there exists at least one fixing of $R_1 = r_1, \cdots, R_m = r_m$ such that

$$|\mathsf{Supp}(E(X_1, r_1), \cdots, E(X_m, r_m))| \geq (1-\varepsilon)2^{dm},$$

then we say that the function E is a *strong* (k, ε, d, s) block min-entropy disperser.

In this paper, we make connections between the block min-entropy disperser and (quasi-)sunflower structures. Formally, we prove the following theorem.

▶ **Theorem 8.** *Suppose that there exists a strong $(k, 0, d, s)$ block min-entropy disperser, $E : \{0,1\}^n \times \{0,1\}^s \to \{0,1\}^d$ for any (w, n, k) block min-entropy source. Then the following holds.*

Let $\mathcal{F}$ be a family of sets where each set has size w. Assume that $|\mathcal{F}| \geq 2^{(k+2)w}$. Then:

(i) *$\mathcal{F}$ contains a 2^d-sunflower.*

(ii) *$\mathcal{F}$ contains a $\left(p, w(1-p)^{2^d}\right)$-quasi-sunflower.*

Observe that the seed length s of the extractor does not play a part in the conclusion of Theorem 8. We then show that we can construct strong block min-entropy extractors and strong zero-error block min-entropy dispersers. Specifically, we have the following theorem.

▶ **Theorem 9.** *There is a constant $c > 1$ such that for any $m, n, k \in \mathsf{N}$ with $k \geq c \log m$, we have:*

- *There is an explicit strong (k, ε, d, s) block min-entropy extractor $E : \{0,1\}^n \times \{0,1\}^s \to \{0,1\}^d$ for (m, n, k) block min-entropy sources, where $s = n$, $d = k/c$ and $\varepsilon = 2^{-\Omega(k)}$.*
- *There is an explicit strong $(k, 0, d, s)$ block min-entropy disperser $E : \{0,1\}^n \times \{0,1\}^s \to \{0,1\}^d$ for (m, n, k) block min-entropy sources, where $s = n$, $d = k/c$.*

Combined with Theorem 8, this gives the sunflower theorem and the quasi-sunflower lemma.

▶ **Corollary 10** (Sunflower theorem, this paper)**.** *There is a constant $c > 1$ such that for any family of sets $\mathcal{F}$ each of size w and any $r > 1$, if $|\mathcal{F}| \geq (wr)^{cw}$, then $\mathcal{F}$ contains a r-sunflower.*

▶ **Corollary 11** (Quasi-sunflower lemma, this paper)**.** *There is a constant $c > 1$ such that for any family of sets $\mathcal{F}$ each of size w, if $|\mathcal{F}| \geq 2^{2w} \cdot \left(\frac{w + \log(1/\gamma)}{p}\right)^{cw}$, then $\mathcal{F}$ contains a (p, γ)-quasi-sunflower.*

1.2 Overview of the techniques

Our reduction from sunflower/quasi-sunflower problems to block min-entropy dispersers is as follows. Suppose the family $\mathcal{F} \subseteq \mathcal{P}(X)$ for some set X, where each set in $\mathcal{F}$ has size w. We first show that without loss of generality we can assume $\mathcal{F}$ has a *normal form*.

▶ **Definition 12** (Normal form). Let X be a finite set and let $\mathcal{F} = \{U_i\}_{i \in I}$ be a family of subsets of X. We say that $\mathcal{F}$ is w-normal if

- For each $U \in \mathcal{F}$, the size of U is w.
- There is a disjoint partition $X_1, \cdots, X_w$ of X such that for every $U \in \mathcal{F}$, we have $|X_j \cap U| = 1$ for each $j \in [w]$.

Consider the uniform distribution over $\mathcal{F}$ of a normal form. There are two possible cases:

Case 1: there is a subset S which appears in many sets of $\mathcal{F}$, that is

$$\big|\{U \in \mathcal{F} : S \subseteq U\}\big| \geq \big|\mathcal{F}\big|/\kappa^{|S|},$$

where κ is a parameter to be determined.

Case 2: every set S does not appear in too many sets of $\mathcal{F}$.

In case 1, S is already like a core in a sunflower or quasi-sunflower, thus we can apply induction on the sub-family $\mathcal{F}_S := \{U \setminus S : (U \in \mathcal{F}) \wedge (S \subseteq U)\}$. In case 2, the condition basically implies that the distribution is relatively flat, which equivalently translates into a block min-entropy source as we defined above. One can naturally imagine that the worst case situation here is that the distribution is actually the uniform distribution over $X_1 \times \cdots \times X_w$, and we show that indeed this is the case by using our zero-error block min-entropy disperser. It is then easy to see that in the worst case, the empty set is a quasi-sunflower, or one can choose a sunflower with size 2^d (the support size in the output of the disperser) whose core is the empty set.

1.3 The role of extractors in our reduction

One can view the block min-entropy extractor/disperser used in our reduction as a gadget, which reduces the sunflower/quasi-sunflower problem in the general case to the much easier case of a uniform distribution (or full support) on $X_1 \times \cdots \times X_w$. This is similar to the role of extractors in recent works that showed lifting theorems from query complexity to communication complexity [11], and linear programming lower bounds for constraint satisfaction problems [14].

In fact, the extractors used in these works are essentially the same as the extractors used in this work (although in this work we need to show that the extractor/disperser is strong, while in [11] and [14] this is not necessary), and the barriers for further improvement are also similar. Specifically, in all such constructions one needs the min-entropy $k \geq c \log m$ for some constant $c > 1$, where m is equal to the size of the sets (i.e., w) in our applications. It is unknown if this dependence on m is necessary for a block min-entropy extractor/disperser to exist. If one can remove the dependence of k on m (even at the price of decreasing the output of the extractor/disperser), then our reduction will give improved bounds for both the sunflower problem and the quasi-sunflower problem. In particular, by Theorem 8 we will be able to show that any family $\mathcal{F}$ of subsets with $|\mathcal{F}| \geq (g(r))^w$ contains a r-sunflower where $g(r)$ is a function on r, and thus prove the sunflower conjecture. It may also lead to a bound of $(O(\log(1/\gamma)/p))^w$ for $\mathcal{F}$ to contain a (p, γ)-quasi-sunflower, and thus improving the DNF sparsification in [12]. Similarly, removing such a dependence will lead to further

improvements in lifting theorems and linear programming lower bounds, as shown in [11] and [14]. In conclusion, we believe that the study of block min-entropy extractors is an important question that needs further investigation.

1.4 Further discussions

Discussions about the sunflower conjecture

In this paper, we show that for any set system $\mathcal{F}$ of size $|\mathcal{F}| \geq w^{cw}$ for some constant $c > 1$, it contains a 3-sunflower. Furthermore, we show that any set system $\mathcal{F}$ with the following Lipschitz condition, must contain three pairwise disjoint sets.

▶ **Definition 13** (Lipschitz condition)**.** Given a collection of sets $\mathcal{F}$ and $r > 0$. We say it is r-Lipschitz, if for any subset S,

$$\big|\{U \in \mathcal{F} : S \subseteq U\}\big| \leq \big|\mathcal{F}\big|/r^{|S|}.$$

▶ **Corollary 14.** *There is a constant $c > 0$, such that for any w-normal set system $\mathcal{F}$, if $\mathcal{F}$ is w^c-Lipschitz then it contains w pairwise disjoint sets.*

This w^c-Lipschitz condition actually comes from the requirement of our disperser that $k \geq c \log w$. As discussed above, it is interesting to ask whether this is necessary. In particular, there may be a way to improve this corollary without using dispersers. We make the following conjecture, which implies the sunflower conjecture.

▶ **Conjecture 15** (Disjoint sets conjecture)**.** *For any $r \geq 3$, there exists a constant $c_r > 1$, such that for any set system $\mathcal{F}$, if $\mathcal{F}$ is c_r-Lipschitz then it contains r pairwise disjoint sets.*

As the disjoint sets conjecture seems hard (it implies the sunflower conjecture), we also make the following simpler conjecture, which is of independent interest.

▶ **Conjecture 16** (2-disjoint sets conjecture)**.** *There exists a constant $c > 1$, such that for any set system $\mathcal{F}$, if $\mathcal{F}$ is c-Lipschitz then it contains 2 pairwise disjoint sets.*

Discussions about quasi-sunflowers

In this paper, we also study quasi-sunflower structures. In particular, we have the following corollary. Below, we use the notation $O_{p,\gamma}(\cdot)$ to hide the specific dependency on the parameters p, γ, which is of less interest to us.

▶ **Corollary 17.** *There is a constant $c > 0$ such that, for any w-normal family $\mathcal{F}$, if $\mathcal{F}$ is r-Lipschitz where $r = (O_{p,\gamma}(w))^c$, then the empty set is a (p, γ)-quasi-sunflower for $\mathcal{F}$.*

It seems the corollary can be further improved. We make the following conjecture.

▶ **Conjecture 18.** *There is a constant $c > 0$ such that, for any w-normal family $\mathcal{F}$, if $\mathcal{F}$ is r-Lipschitz where $r = (O_{p,\gamma}(\log w))^c$, then the empty set is a (p, γ)-quasi-sunflower for $\mathcal{F}$.*

The reason for the $\log w$ term is the following example, which we believe is the worst instance for quasi-sunflower structures. Fix $p = \gamma = 1/2$ for convenience. Let $X_1, \ldots, X_w$ be w disjoint sets each of size $c \log w$ for some small enough $c > 0$. Define the collection of sets as $\mathcal{F} := X_1 \times \cdots \times X_w$. Then $\mathcal{F}$ does not contain a (p, γ)-quasi-sunflower.

We note that proving our conjectures, or even improving our corollaries will lead to interesting improvements on the sunflower theorem or the quasi-sunflower lemma. This will in turn lead to improvements in other applications such as DNF sparsification, constructing pseudorandom generators for DNF formulas, and approximate counting the number of satisfying assignments for DNF formulas.

2 Preliminaries

We first review some basic definitions in probability.

▶ **Definition 19.** Let D be a distribution over a sample space X. Its entropy is

$$\mathcal{H}(D) = \sum_x \Pr[D = x] \cdot \log\left(\frac{1}{\Pr[D = x]}\right).$$

Its min-entropy is

$$\mathcal{H}_\infty(D) = \min_x \left\{\log\left(\frac{1}{\Pr[D = x]}\right)\right\}.$$

Its max-entropy is

$$\mathcal{H}_0(D) = \log|\{\mathsf{Supp}(D)\}|.$$

▶ **Definition 20** (Statistical distance)**.** Let D_0 and D_1 be distributions over a finite sample space X. The statistical distance between D_0 and D_1 is defined as

$$\mathrm{dist}(D_0, D_1) = \frac{1}{2}\sum_{x\in X}\big|\Pr[D_0 = x] - \Pr[D_1 = 1]\big|.$$

3 A construction of a block min-entropy extractor

We use the following well-known extractor based on the inner product function [5]. We denote by $\mathbb{F}_q$ the finite field on q elements. When $q = 2^\ell$ we identify $\mathbb{F}_q$ with $\{0,1\}^\ell$ and $\mathbb{F}_q^t$ with $\{0,1\}^{t\ell}$.

▶ **Theorem 21** ([5])**.** *Let $t, \ell \geq 1$ and take $q = 2^\ell, n = t\ell$. Let X, Y be independent sources on $\mathbb{F}_q^t \cong \{0,1\}^n$ with min-entropy k_1, k_2 respectively. Let* IP *be the inner product function over the field $\mathbb{F}_q$. Then:*

$$\mathrm{dist}\left((\mathrm{IP}(X,Y), X), (U_\ell, X)\right) \leq \varepsilon \quad \textit{and} \quad \mathrm{dist}\left((\mathrm{IP}(X,Y), Y), (U_\ell, Y)\right) \leq \varepsilon$$

where $\varepsilon = 2^{\frac{-(k_1+k_2-n-\ell)}{2}}$.

Now we can construct a block min-entropy extractor as follows. Given parameters n, k, choose a field $\mathbb{F}_q$ such that $q = 2^\ell$ with $\ell = \alpha k$ for some constant $0 < \alpha < 1$ to be determined later. Without loss of generality we assume that $n = \ell t$ for some integer t. We view $X \in \{0,1\}^n$ as a vector in $\mathbb{F}_q^t$ and choose a uniform independent seed $R \in \{0,1\}^n \cong \mathbb{F}_q^t$.

A block min-entropy extractor

1. Given parameters m, n, k let q, t be as described above.
2. Sample $(x_1, \ldots, x_m)$ from the block min-entropy distribution $X = (X_1, \ldots, X_m) \in (\mathbb{F}_q^t)^m$.
3. Uniformly sample $(R_1, \ldots, R_m) \in (\mathbb{F}_q^t)^m$.
4. Output $Z := (\mathrm{IP}(x_1, R_1), \ldots, \mathrm{IP}(x_m, R_m))$.

We are now ready to prove the following theorem.

Theorem 9 (restated). *Let $X = (X_1, \cdots, X_m)$ be an (m, n, k) block min-entropy source. Let $Z \in \{0,1\}^{\ell m}$ be the output of the above block min-entropy extractor applied to X. There exists a constant $c > 1$ such that if $k \geq c \log m$, then the following holds for any error $\varepsilon = 2^{-\Omega(k)}$:*

- *With probability $1 - \varepsilon$ over the fixing of the seed $(R_1, \ldots, R_m)$,*

$$\left|\Pr[Z = z] - 2^{-\ell m}\right| \leq \varepsilon \cdot 2^{-\ell m} \qquad \forall z \in \{0,1\}^{\ell m}.$$

  In particular, in such cases $\mathcal{H}_0(Z) = \ell m$
- $\mathrm{dist}\left((Z, R_1, \cdots, R_m), (U, R_1, \cdots, R_m)\right) \leq 2\varepsilon$.

Proof. Note that we have a joint distribution $(X_1, \cdots, X_m)$ that has block min-entropy k. The output of the local extractor applied to $(X_1, \cdots, X_m)$, using m independent uniform seeds $(R_1, \cdots, R_m)$, is a distribution $(Z_1, \cdots, Z_m)$ over $\{0,1\}^{\ell m} = \mathbb{F}_q^m$ where $Z_i = \mathrm{IP}(X_i, R_i)$ for each i.

For any fixing of the seed $(R_1 = r_1, \cdots, R_m = r_m)$, the distribution $(Z_1, \cdots, Z_m)$ is a deterministic function of $(X_1, \cdots, X_m)$, and we will view this distribution as a function $\mathcal{D} : \{0,1\}^{\ell m} \to [0,1]$ where the image of each input is its associated probability in the distribution. We now write this function in its Fourier basis:

$$\mathcal{D}(z) = \sum_{S \subseteq [\ell m]} \hat{\mathcal{D}}(S) \chi_S(z),$$

where $z = (z_1, \cdots, z_m) \in \{0,1\}^{\ell m}$, $\chi_S(z) = (-1)^{\sum_{i \in S} z(i)} \in \{+1, -1\}$, and

$$\hat{\mathcal{D}}(S) = 2^{-\ell m} \cdot \sum_z \mathcal{D}(z)\chi_S(z) = 2^{-\ell m} \cdot \mathbb{E}_{z \sim \mathcal{D}}[\chi_S(z)].$$

Here we use $z(i)$ to stand for the i'th *bit* of the string z. This is to distinguish between the notation z_i, which referes to the i'th *block* of the string z, that contains ℓ bits.

Note that $\hat{\mathcal{D}}(\emptyset) = 2^{-\ell m}$ since $\mathcal{D}$ is a probability distribution. Thus we have that $\forall z \in \{0,1\}^{\ell m}$,

$$\left|\mathcal{D}(z) - 2^{-\ell m}\right| = \left|\sum_{S \subseteq [\ell m], S \neq \emptyset} \hat{\mathcal{D}}(S)\chi_S(z)\right| \leq \sum_{S \subseteq [\ell m], S \neq \emptyset} \left|\hat{\mathcal{D}}(S)\right|.$$

Note that for any $S \subseteq [\ell m]$, $\chi_S(Z)$ corresponds to the parity of a subset of the bits in Z. For each $Z_j, j \in [m]$, this parity may or may not involve any bits in Z_j. We will be interested in the number of j's such that $\chi_S(Z)$ involves at least one bit from Z_j, and we call this number $\Delta(S)$. Note that $\Delta(\emptyset) = 0$ and $1 \leq \Delta(S) \leq m$ for any $S \neq \emptyset$.

We now have the following lemma.

▶ **Lemma 22.** *If $\Delta(S) = h$, then with probability $1 - 2^{\frac{-h(1-\alpha)k}{4}}$ over the fixing of the seed $(R_1 = r_1, \cdots, R_m = r_m)$, we have that $|\hat{\mathcal{D}}(S)| \leq 2 \cdot 2^{-\ell m} 2^{\frac{-h(1-\alpha)k}{4}}$.*

Proof. Without loss of generality assume that the Z_j's from which $\chi_S(Z)$ involves at least one bit are $(Z_1, \cdots, Z_h)$. Note that for any $Z_i \in \{0,1\}^\ell = \mathbb{F}_q$, any parity of the bits of Z_i corresponds exactly to the first bit of $a \cdot Z_i$ viewed as a vector in $\{0,1\}^\ell$, for some $a \in \mathbb{F}_q$ and the operation $\cdot$ is multiplication in the field $\mathbb{F}_q$. Moreover this correspondence is a bijection in the sense that different parities correspond to different elements $a \in \mathbb{F}_q$. The

special case of parity over the empty set corresponds to the case of $a = 0$. Thus, $\sum_{i\in S} Z(i)$ corresponds to the first bit of $\sum_{j\in[h]} a_j Z_j$ viewed as a vector in $\{0,1\}^\ell$, for some non-zero $\{a_j \in \mathbb{F}_q : j \in [h]\}$. Note that

$$\sum_{j\in[h]} a_j Z_j = \sum_{j\in[h]} a_j \mathrm{IP}(X_j, R_j) = \sum_{j\in[h]} \mathrm{IP}(a_j X_j, R_j) = \mathrm{IP}((a_1X_1, \cdots, a_hX_h), (R_1, \cdots, R_h)).$$

Since each $a_j \neq 0$ the transformation from $(x_1, \cdots, x_h)$ to $(a_1x_1, \cdots, a_hx_h)$ is a bijection. Thus we know the distribution $(a_1X_1, \cdots, a_hX_h)$ has min-entropy kh, while $(R_1, \cdots, R_h)$ has min-entropy nh. Thus by Theorem 21 applied over the field $\mathbb{F}_{2^{\ell h}}$ we have that

$$\mathrm{dist}\left((\sum_{j\in[h]} a_j Z_j, R_1, \cdots, R_m), (U_{\ell h}, R_1, \cdots, R_m)\right) \leq 2^{\frac{-(h(k-\ell))}{2}} = 2^{\frac{-h(1-\alpha)k}{2}}.$$

In particular, as $\chi_S(Z)$ is the first bit of $\sum_{j\in[h]} a_j Z_j$, we have

$$\mathrm{dist}\,(\chi_S(Z), R_1, \cdots, R_m), (U_1, R_1, \cdots, R_m)) \leq 2^{\frac{-h(1-\alpha)k}{2}}.$$

By Markov's inequality this means that with probability $1 - 2^{\frac{-h(1-\alpha)k}{4}}$ over the fixing of the seed $R = (R_1, \ldots, R_m)$, we have $|\hat{\mathcal{D}}(S)| = |2^{-\ell m} \cdot \mathbb{E}_{z\sim\mathcal{D}}[\chi_S(z)]| \leq 2 \cdot 2^{-\ell m} 2^{\frac{-h(1-\alpha)k}{4}}$. ◀

Next, note that the number of S with $\Delta(S) = h$ is $\binom{m}{h}(2^\ell - 1)^h \leq 2^{(\ell+\log m)h}$. Recall that $\ell = \alpha k$ and $k \geq c \log m$. We can choose the constants α, c such that $2^{\ell+\log m} 2^{\frac{-(1-\alpha)k}{4}} \leq 2^{-\frac{k}{8}}$. Now we have as long as $k \geq 8$,

$$\sum_{h=1}^{m} \binom{m}{h} (2^\ell - 1)^h 2^{\frac{-h(1-\alpha)k}{4}} \leq \sum_{h=1}^{m} 2^{-\frac{hk}{8}} \leq 2^{-\frac{k}{8}+1}.$$

Set $\varepsilon = 2^{-\frac{k}{8}+2} = 2^{-\Omega(k)}$. By the union bound we have that with probability at least $1 - \varepsilon$ over the fixing of the seed $(R_1 = r_1, \cdots, R_m = r_m)$, for every $S \neq \emptyset$ with $\Delta(S) = h$, $|\hat{\mathcal{D}}(S)| \leq 2 \cdot 2^{-\ell m} 2^{\frac{-h(1-\alpha)k}{4}}$. Thus for any such seed we have that

$$\left|\mathcal{D}(z) - 2^{-\ell m}\right| \leq \sum_{S\subseteq[\ell m], S\neq\emptyset} \left|\hat{\mathcal{D}}(S)\right| \leq \varepsilon \cdot 2^{-\ell m}.$$

This concludes the proof of the first part of Theorem 9. For the second part, notice that conditioned on the fixing of any seed $R_1, \ldots, R_m$, with probability $1 - \varepsilon$ the statistical distance is at most ε, and otherwise it is trivially bounded by 1. So overall the statistical distance between $(Z, R_1, \cdots, R_m)$ and $(U_{\ell m}, R_1, \cdots, R_m)$ is at most 2ε. ◀

4 Compressing set systems by the block min-entropy extractor

In this section, we focus on the set systems that satisfy the Lipschitz condition, and show a compression operator for such set systems. Our compression is based on the block min-entropy extractor. We first show that it suffices to consider w-normal set systems (see Definition 12).

▶ **Lemma 23.** *Let $\mathcal{F}$ be a family of sets such that each set has size w. Then there exists a w-normal sub-family $\mathcal{F}'$ of $\mathcal{F}$ with $|\mathcal{F}'| \geq |\mathcal{F}|/2^{2w}$.*

Proof. Let $U \in \mathcal{F}$ be a set, and let $X_1, \ldots, X_w$ be a random partition of X. Then

$$\Pr_{X_1,\cdots,X_w}[\forall j \in [w], |U \cap X_j| = 1] = \frac{w!}{w^w}.$$

Then by an average argument, there is a partition $(X_1, \cdots, X_w)$ such that

$$|\{U \in \mathcal{F} : \forall j \in [w], |U \cap X_j| = 1\}| \geq |\mathcal{F}| \cdot \frac{w!}{w^w}$$

The claim then follows since $\frac{w!}{w^w} \geq 2^{-2w}$. ◀

Now we can focus on normal set systems. Given a finite set X, we denote by X_p the distribution over subsets $W \subset X$, where each $x \in X$ appears in W independently with probability p.

▶ **Lemma 24.** *Let $u \geq w$. Let c be the constant from theorem 9. Then for every w-normal set system which is u^c-Lipschitz (recall Definition 13), it holds that*

$$\Pr_{W \sim X_p}[\exists U \in \mathcal{F}, U \subseteq W] \geq 1 - w(1-p)^u.$$

To prove this lemma, we first define a "worst case" instance, and then show that all other instances behave better than this case. Let $X_1^*, \cdots, X_w^*$ be w disjoint sets each of size u. Define the family $\mathcal{U}^*$ as

$$\mathcal{U}^* = \left\{\{x_1, \cdots, x_w\} : \forall j \in [w], x_j \in X_j^*\right\}.$$

▶ **Claim 25.** *Let $\mathcal{U}^*$ as defined above. Then*

$$\Pr_{W \sim X_p}[\exists U \in \mathcal{U}^*, U \subseteq W] \geq 1 - w(1-p)^u.$$

Proof. By the definition of $\mathcal{U}^*$, we have that

$$\begin{aligned}\Pr_W[\forall U \in \mathcal{U}^*, U \not\subseteq W] =& \Pr_W[\exists j \in [w], X_j \cap W = \emptyset] \\ \leq& \sum_{j \in [w]} \Pr_W[X_j \cap W = \emptyset] \\ =& w(1-p)^u.\end{aligned}$$

◀

Let X, Y be finite sets, $h : X \to Y$ a map. Given a set $U \subset X$ define $h = \{h(x) : x \in U\} \subset Y$. Given a family $\mathcal{F} \subseteq \mathcal{P}(X)$ define $h(\mathcal{F}) \subseteq \mathcal{P}(Y)$ as

$$h(\mathcal{F}) = \{h : U \in \mathcal{F} \text{ and } h \text{ is injective on } U\}.$$

▶ **Lemma 26.** *Let X and Y be sets, $h : X \to Y$ a map, $\mathcal{F} \subset \mathcal{P}(X)$. Then*

$$\Pr_{W_Y \sim Y_p}[\exists U \in h(\mathcal{F}), U \subseteq W_Y] \leq \Pr_{W_X \sim X_p}[\exists U \in \mathcal{F}, U \subseteq W_X].$$

Proof. Without loss of generality, we can assume the map h is surjective, because elements $y \in Y \setminus h(X)$ do not affect the events. If $|Y| = |X|$ then h is a bijection and hence $\mathcal{F}$ and $h(\mathcal{F})$ are the same, up to renaming the elements. So, assume $|Y| < |X|$. It suffices to prove the lemma for the case that $|Y| = |X| - 1$, as the general case follows from applying this case iteratively (namely, decompose h as a sequence of maps, each reduces the domain size by one).

So, assume $|Y| = |X| - 1$. In this case, there is a unique pair $x_1, x_2 \in X$ such that $h(x_1) = h(x_2) = y$. We may assume without loss of generality (by renaming the elements of Y) that h is the identity map on $X' = X \setminus \{x_1, x_2\}$. This allows us to jointly sample (W_X, W_Y) as follows. Sample $W' \sim X'_p, W'_X \sim \{x_1, x_2\}_p, W'_Y \sim \{y\}_p$ and set $W_X = W' \cup W'_X, W_Y = W' \cup W'_Y$. We will show that for every fixed $W' = w'$,

$$\Pr_{W_Y \sim Y_p}[\exists U \in h(\mathcal{F}), U \subseteq W_Y \mid W' = w'] \leq \Pr_{W_X \sim X_p}[\exists U \in \mathcal{F}, U \subseteq W_X \mid W' = w']. \tag{1}$$

The lemma then follows by averaging over W'.

To that end, fix W'. Let $\mathcal{F}' = \{U \setminus X' : U \in \mathcal{F}, (U \cap X') \subset W'\}$. Note that $\mathcal{F}' \subseteq \mathcal{P}(\{x_1, x_2\})$. Similarly, define $\mathcal{F}'' = \{U \setminus X' : U \in \langle(\mathcal{F}), (U \cap X') \subset W'\}$. Note that $\mathcal{F}'' \subseteq \mathcal{P}(\{y\})$. Equation (1) is equivalent to

$$\Pr_{W'_Y \sim \{y\}_p}[\exists U \in \mathcal{F}'', U \subseteq W'_Y] \leq \Pr_{W'_X \sim \{x_1, x_2\}_p}[\exists U \in \mathcal{F}', U \subseteq W'_X]. \tag{2}$$

We verify Equation (2) by a case analysis.

- **(i)** If $\mathcal{F}''$ is empty then the LHS of Equation (2) is 0, while the RHS is non-negative.
- **(ii)** If $\emptyset \in \mathcal{F}''$ then $\emptyset \in \mathcal{F}'$. In this case, both the LHS and RHS of Equation (2) equal 1.
- **(iii)** If $\mathcal{F}'' = \{\{y\}\}$ then either $\{x_1\} \in \mathcal{F}'$ or $\{x_2\} \in \mathcal{F}'$. In either case, the LHS of Equation (2) equals p, while the RHS is at least p. ◀

We now prove Lemma 24. Let $\mathcal{F}$ be a family of sets that satisfies the assumptions. We will show there is a function h such that $h(\mathcal{F}) = \mathcal{U}^*$. The extractor from 9, with an appropriate choice of seed, provides such a function h.

Proof of Lemme 24. Let $\mathcal{F}$ be a w-normal family of sets that satisfies the Lipchitz condition. We first define the function h. Since $\mathcal{F}$ is a w-normal set, there exists a partition of X to $X_1, \ldots, X_w$ such that for each $U \in \mathcal{F}$ and $j \in [w]$, $|X_j \cap U| = 1$.

Define the sample space as $X_1 \times \cdots \times X_w$. With a slight abuse to use the notations, we identify $\mathcal{F} \subseteq \mathcal{P}(X_1 \times \cdots \times X_w)$, and let D be a uniform distribution over $\mathcal{F}$. Since $\mathcal{F}$ is u^c-Lipschitz, the distribution D is a $(w, \log X, k)$ block min-entropy source with $k = c \log u \geq c \log w$. Then by Theorem 9, there exists seeds $r_1, \ldots, r_w$ such that $(\mathrm{IP}(D_1, r_1), \ldots, \mathrm{IP}(D_w, r_w))$ has full support, where $D = (D_1, \ldots, D_w)$. Note that the output of $\mathrm{IP}(\cdot, \cdot)$ is in $\{0,1\}^{k/c} \cong [u]$. We can now define h as follows:

$$h(x) = (\mathrm{IP}(x, r_j), j) \qquad \forall x \in X_j.$$

Note that by definition, h is injective on any $U \in X_1 \times \cdots \times X_w$. We identify elements of $\mathcal{U}^*$ with $\{(a_1, 1), \ldots, (a_w, w)\}$ with $a_i \in [u]$. Thus $h(\mathcal{F}) = \mathcal{U}^*$. The lemma now follows from Lemma 26 and Claim 25. ◀

We will also need the following lemma.

▶ **Lemma 27.** *Let $u \geq w$. Let c be the constant from theorem 9. Then for every w-normal set system $\mathcal{F}$ which is u^c-Lipschitz (recall Definition 13), it holds that $\mathcal{F}$ contains u pairwise disjoint sets.*

Proof. The proof is very similar to the proof of Lemma 24. There is a map h for which $h(\mathcal{F}) = \mathcal{U}^*$. Note that $\mathcal{U}^*$ contains u pairwise disjoint sets, $U'_1, \ldots, U'_u$. By definition, $U'_i = h(U_i)$. But then also $U_1, \ldots, U_u$ must be pairwise disjoint. ◀

4.1 Sunflowers and quasi-sunflowers from compression

Now we can prove Theorem 8.

Theorem 8 (restated). *Suppose that there exists a strong $(k, 0, d, s)$-block min-entropy disperser, $E : \{0,1\}^n \times \{0,1\}^s \to \{0,1\}^d$ for any (w, n, k)-block min-entropy source. Then the following holds.*

Let $\mathcal{F}$ be a family of sets where each set has size w. Assume that $|\mathcal{F}| \geq 2^{(k+2)w}$. Then:

(i) *$\mathcal{F}$ contains a 2^d-sunflower.*

(ii) *$\mathcal{F}$ contains a $\left(p, w(1-p)^{2^d}\right)$-quasi-sunflower.*

Proof. By Lemma 23, there is a w-normal subclass $\mathcal{F}' \subseteq \mathcal{F}$ of size $|\mathcal{F}'| \geq 2^{kw}$. There are two possible cases.

Case 1: There is a subset $S \subseteq X$ such that

$$|\{U \in \mathcal{F}' : S \subseteq U\}| \geq |\mathcal{F}'| \cdot 2^{-k|S|}.$$

Define the family $\mathcal{F}'_S := \{U \setminus S : (U \in \mathcal{F}') \wedge (S \subseteq U)\}$. Notice that

- $\mathcal{F}'_S$ is $(w - |S|)$-normal.
- $|\mathcal{F}'_S| \geq |\mathcal{F}'| \cdot 2^{-k|S|} \geq 2^{k(w-|S|)}$.

By induction both (i) and (ii) hold.

Case 2: For all $S \subseteq X$,

$$|\{U \in \mathcal{F}' : S \subseteq U\}| \leq |\mathcal{F}'| \cdot 2^{-k|S|}$$

Notice that this is the Lipchitz condition for Lemma 24 and Lemma 27. Their conclusions are precisely (i) and (ii). ◀

References

1 Noga Alon, Amir Shpilka, and Christopher Umans. On sunflowers and matrix multiplication. *Computational Complexity*, pages 214–223, 2012.

2 Boaz Barak, Guy Kindler, Ronen Shaltiel, Benny Sudakov, and Avi Wigderson. Simulating independence: New constructions of condensers, ramsey graphs, dispersers, and extractors. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 1–10. ACM, 2005.

3 Avraham Ben-Aroya, Dean Doron, and Amnon Ta-Shma. Explicit two-source extractors for near-logarithmic min-entropy. In *Proceedings of the 49th Annual ACM Symposium on Theory of Computing*, 2017.

4 Eshan Chattopadhyay and David Zuckerman. Explicit two-source extractors and resilient functions. In *Proceedings of the 48th Annual ACM Symposium on Theory of Computing*, 2016.

5 Benny Chor and Oded Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM Journal on Computing*, 17(2):230–261, 1988.

6 Gil Cohen. Two-source extractors for quasi-logarithmic min-entropy and improved privacy amplification protocols. In *Proceedings of the 49th Annual ACM Symposium on Theory of Computing*, 2017.

7 Ernie Croot, Vsevolod F. Lev, and Péter Pál Pach. Progression-free sets in z_4^n are exponentially small. *Annals of Mathematics*, 185(1):331–337, 2017.

8 Paul Erdős. Some remarks on the theory of graphs. *Bull. Amer. Math. Soc.*, 53(4):292–294, 1947.

9 Paul Erdős and R Rado. Intersection theorems for systems of sets. *Journal of the London Mathematical Society*, 35(1):85–90, 1960.

10 Paul Erdős and George Szekeres. A combinatorial problem in geometry. *Compositio Mathematica*, 2:463–470, 1935.

11 M. Goos, S. Lovett, R. Meka, T. Watson, and D. Zuckerman. Rectangles are nonnegative juntas. *SIAM Journal on Computing*, 45(5):1835–1869, 2016.

12 Parikshit Gopalan, Raghu Meka, and Omer Reingold. Dnf sparsification and a faster deterministic counting algorithm. *computational complexity*, 22(2):275–310, 2013.

13 Ben Green and Terence Tao. The primes contain arbitrarily long arithmetic progressions. *Annals of Mathematics*, 167(2):481–547, 2008.

14 Pravesh K. Kothari, Raghu Meka, and Prasad Raghavendra. Approximating rectangles by juntas and weakly-exponential lower bounds for lp relaxations of csps. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, 2017.

15 Xin Li. Three source extractors for polylogarithmic min-entropy. In *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science*, 2015.

16 Xin Li. Improved non-malleable extractors, non-malleable codes and independent source extractors. In *Proceedings of the 49th Annual ACM Symposium on Theory of Computing*, 2017.

17 Alexander Razborov. Some lower bounds for the monotone complexity of some boolean functions. *Soviet Math. Dokl.*, 31:354–357, 1985.

18 Benjamin Rossman. The monotone complexity of k-clique on random graphs. In *Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 193–201. IEEE Computer Society, 2010.

19 Jordan S. Ellenberg and Dion Gijswijt. On large subsets of f_q^n with no three-term arithmetic progression. *Annals of Mathematics*, 185(1):339–343, 2017.

20 Endre Szemerédi. On sets of integers containing no k elements in arithmetic progression. *Acta Arithmetica*, 27:199–245, 1975.

21 Endre Szemerédi. Regular partitions of graphs. *Problèmes combinatoires et théorie des graphes*, 260:399–401, 1978.

Torpid Mixing of Markov Chains for the Six-vertex Model on $\mathbb{Z}^2$

Tianyu Liu
University of Wisconsin-Madison, Madison, WI, USA
tl@cs.wisc.edu

Abstract

In this paper, we study the mixing time of two widely used Markov chain algorithms for the six-vertex model, Glauber dynamics and the directed-loop algorithm, on the square lattice $\mathbb{Z}^2$. We prove, for the first time that, on finite regions of the square lattice these Markov chains are torpidly mixing under parameter settings in the ferroelectric phase and the anti-ferroelectric phase.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms, Theory of computation → Random walks and Markov chains

Keywords and phrases the six-vertex model, Eulerian orientations, square lattice, torpid mixing

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.52

Funding This work was supported by NSF CCF-1714275.

Acknowledgements The author thanks Professor Jin-Yi Cai for the valuable suggestions on the preliminary version of this paper.

1 Introduction

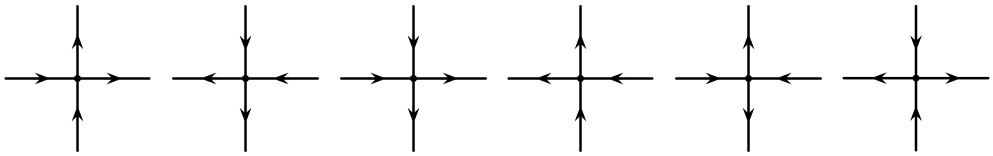
Introduced by Linus Pauling [23] in 1935 to describe the properties of ice, the *six-vertex model* or the *ice-type model* was originally studied in statistical mechanics as an abstraction of crystal lattices with hydrogen bonds. During the following decades, it has attracted enormous interest in many disciplines of science, and become one of the most fundamental models defined on the square lattice. In particular, the discovery of *integrability* of the six-vertex models with periodic boundary conditions was considered a milestone in statistical physics [16, 14, 15, 29, 10].

For computational expediency and modeling purposes, physicists almost entirely focused on planar lattice models. On the square lattice $\mathbb{Z}^2$, every vertex is connected by an edge to four "nearest neighbors". States of the six-vertex model on $\mathbb{Z}^2$ are orientations of the edges on the lattice satisfying the *ice-rule* — every vertex has two incoming edges and two outgoing edges, i.e., they are *Eulerian orientations*. The name of six-vertex model comes from the fact that there are six ways of arranging directions of the edges around a vertex (see Figure 1).

In general, each of the six local arrangements will have a weight, denoted by $w_1, \ldots, w_6$, using the ordering of Figure 1. The total weight of a state is the product of all vertex weights in the state. If there is no ambient electric field, by physical considerations, then the total weight of a state should remain unchanged when flipping all arrows [3]. Thus one may assume without loss of generality that $w_1 = w_2 = a, w_3 = w_4 = b, w_5 = w_6 = c$. This complementary invariance is known as *arrow reversal symmetry* or *zero field assumption*. In this paper, we assume $a, b, c > 0$, as is the case in *classical* physics. We study the six-vertex model restricted to a finite region of the square lattice with various boundary conditions customarily studied in statistical physics literature. On a finite subset $\Lambda \subset \mathbb{Z}^2$, denote the

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018).
Editors: Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer; Article No. 52; pp. 52:1–52:15

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



(a) 1 **(b)** 2 **(c)** 3 **(d)** 4 **(e)** 5 **(f)** 6

Figure 1 Valid configurations of the six-vertex model.

set of valid configurations (i.e. Eulerian orientations) by Ω. The probability that the system is in a state $\tau \in \Omega$ is given by the Gibbs distribution

$$\mu(\tau) = \frac{1}{Z}\left(a^{n_1+n_2} b^{n_3+n_4} c^{n_5+n_6}\right),$$

where n_i is the number of vertices in type i $(1 \le i \le 6)$ on Λ in the state τ, and the partition function Z is a normalizing constant which is the sum of the weights of all states.

In 1967, Elliot Lieb [16] famously showed that, for parameters $(a, b, c) = (1, 1, 1)$ on the square lattice graph, as the side N of the square approaches ∞, the value of the "partition function per vertex" $W = Z^{1/N^2}$ approaches $\left(\frac{4}{3}\right)^{3/2} \approx 1.5396007\ldots$ (this is called Lieb's square ice constant). This result is called an exact solution of the model, and is considered a triumph. After that, exact solutions for other parameter settings have been obtained in the limiting sense [14, 15, 29, 10]. Readers are referred to [7] for known results in the computational complexity of (both exactly and approximately) computing the partition function Z of the six-vertex model on *general* 4-regular graphs.

In statistical physics, *Markov chain Monte Carlo (MCMC)* is the most popular tool to numerically study the properties of the six-vertex model. A partial list includes [25, 31, 2, 9, 30, 1, 19]. In the literature, two Markov chain algorithms are mainly used. The first one is *Glauber dynamics*. It can be shown that there is a correspondence between Eulerian orientations of the *edges* and proper three-colorings of the *faces* on a rectangle region of the square lattice. (See Chapter 8 of [3] for a proof). Therefore, the Glauber dynamics for the three-coloring problem on square lattice regions (which changes a local color at each step) can be employed to sample Eulerian orientations. In fact, this simple Markov chain is used in numerical studies (e.g. in [9, 1, 19] for the density profile) of the six-vertex model under various boundary conditions. The second one is the *directed-loop algorithm*. Invented by Rahman and Stillinger [25] and widely adopted in the literature (e.g., [31, 2, 30]), the transitions of this algorithm are composed of creating, shifting, and merging of two "defects" on the edges. An interesting aspect is that this process depicts the *Bjerrum defects* happening in real ice [2]. More detailed descriptions of the two Markov chain algorithms can be found in Section 2.

With the heavy usage of MCMC in statistical mechanics for the six-vertex model, the efficiency of Markov chain algorithms was inevitably brought into focus by physicists. Many of them (e.g. [2, 30, 19]) reported that Glauber dynamics and the directed-loop algorithms of the six-vertex model experienced significant slowdown and are even "impractical" for simulation purposes when the parameter settings are in the *ordered phases* (see Figure 2a, in the regions FE & AFE). Despite the concern and numerical experience for the convergence rate of these algorithms, there is no previous provable result except for one point (that corresponds to the unweighted case) in the parameter space. This is in stark contrast to the popular studies on the mixing rate of Markov chains for the ferromagnetic Ising model [20, 21, 8, 17] and hardcore gas model on lattice regions [5, 26, 4].

Prior to [7], to our best knowledge, the only provable result in the complexity of approximate sampling and counting for the six-vertex model is at the single, *unweighted*, parameter setting $(a, b, c) = (1, 1, 1)$ where the partition function counts Eulerian orientations. In the unweighted case, all known results are positive. Mihail and Winkler's pioneering work [22] gave the first *fully polynomial randomized approximation scheme (FPRAS)* for the number of Eulerian orientations on a general graph (not necessarily 4-regular). Luby, Randall, and Sinclair showed that Glauber dynamics with extra moves is rapidly mixing on rectangular regions of the square lattice with fixed boundary conditions [18]. Randall and Tetali proved the rapid mixing of the Glauber dynamics (without extra moves) with fixed boundary conditions by a comparison technique applied to this Markov chain and the Luby-Randall-Sinclair chain [27]. Goldberg, Martin, and Paterson extended further the rapid mixing of Glauber dynamics to the free-boundary case [11]. The unweighted setting is the single green point depicted in the blue region of Figure 2b.

In [7], Cai, Liu, and Lu showed that under parameter settings (a, b, c) with $a^2 \leq b^2 + c^2$, $b^2 \leq a^2 + c^2$, and $c^2 \leq a^2 + b^2$ (the blue region in Figure 2b), the directed-loop algorithm mixes in polynomial time with regard to the size of input for any general 4-regular graph, resulting in an FPRAS for the partition function of the six-vertex model. Moreover, it is shown that in the ordered phases (FE & AFE in Figure 2a), the partition function on a general graph is not efficiently approximable unless NP=RP. Although the rapid mixing property for the directed-loop algorithm on general 4-regular graphs implies the same on the lattice region, the hardness result for general 4-regular graphs has no implications on the mixing rate of Markov chains for the six-vertex model on the square lattice in the ordered phases (FE & AFE).

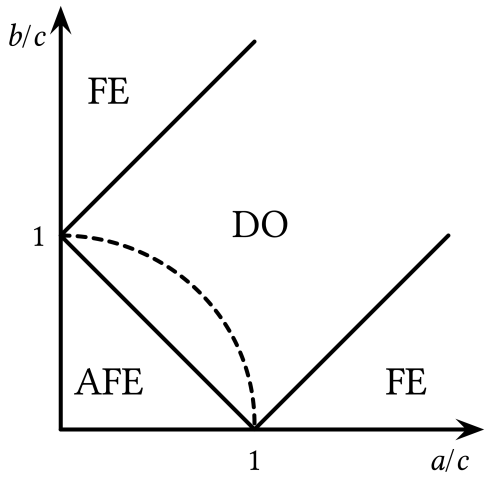
In this paper, we give the first provable negative results on mixing rates of the two Markov chains for the six-vertex model under parameter settings in the ferroelectric phases and the anti-ferroelectric phase. Our results conform to the phase transition phenomena in physics. Here we briefly describe the phenomenon of phase transition of the zero-field six-vertex model (see Baxter's book [3] for more details). On the square lattice in the thermodynamic limit: (1) When $a > b + c$ (FE: ferroelectric phase) any finite region *tends to* be frozen into one of the two configurations where either all arrows point up or to the right (Figure 1-1), or all point down or to the left (Figure 1-2). (2) Symmetrically when $b > a + c$ (also FE) all arrows point down or to the right (Figure 1-3), or all point up or to the left (Figure 1-4). (3) When $c > a + b$ (AFE: anti-ferroelectric phase) configurations in Figure 1-5 and Figure 1-6 alternate. (4) When $c < a + b$, $b < a + c$, and $a < b + c$, the system is disordered (DO: disordered phase) in the sense that all correlations decay to zero with increasing distance; in particular on the dashed curve $c^2 = a^2 + b^2$ the model can be solved by Pfaffians exactly [10], and the correlations decay inverse polynomially, rather than exponentially, in distance. See Figure 2a.

Let Λ be a square region on the square lattice. We show the following two theorems.

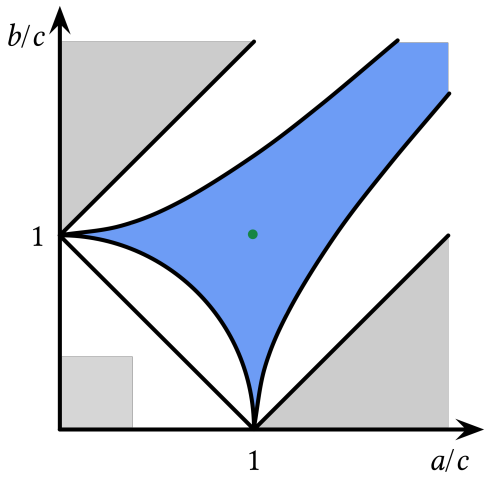
▶ **Theorem 1.1** (Ferroelectric phase). *The directed-loop algorithm for the six-vertex model under parameter settings (a, b, c) with $a > b + c$ or $b > a + c$ (i.e. the whole FE) mixes torpidly on Λ with periodic boundary conditions.*

▶ Remark. We note that for *periodic boundary conditions* Glauber dynamics is *not* irreducible, so we do not consider that.

▶ **Theorem 1.2** (Anti-ferroelectric phase). *Both Glauber dynamics and the directed-loop algorithm for the six-vertex model under parameter settings (a, b, c) with $c \geq 2.639 \max(a, b)$ (in AFE) mix torpidly on Λ with free boundary conditions and periodic boundary conditions.*



(a) Phase diagram of the six-vertex model.



(b) Mixing time of Markov chains for the six-vertex model on $\mathbb{Z}^2$.

Figure 2

Parameter settings covered by the above two theorems are depicted as the grey region in Figure 2b. Given that the F *model* in statistical mechanics is a special case of the six-vertex model when $a = b = 1$ [14], Theorem 1.2 holds for the F model with $c \geq 2.639$.

Our proofs build on the equivalence between small conductance and torpid mixing by Jerrum and Sinclair [28]. When arguing Markov chains for the six-vertex model in the anti-ferroelectric phase have small conductance, we switch our view between finite regions of the square lattice and their *medial graphs*. This transposition allows us to adopt a *Peierls argument* which has been used in statistical physics to prove the existence of phase transitions (e.g., [24, 6]), and in theoretical computer science to prove the torpid mixing of Markov chains (e.g., [26, 4]).

In the proof of Theorem 1.2, we introduce a version of the *fault line argument* for the six-vertex model. Fault line arguments are introduced by Dana Randall [26] for the lattice hardcore gas and latter adapted in [13] for the lattice ferromagnetic Ising, which proves torpid mixing of Markov chains via topological obstructions. The constant 2.639 comes from an upper bound for the connective constant for the square lattice self-avoiding walks [12].

2 Preliminaries

2.1 Markov chains

2.1.1 Glauber dynamics

Denote by Λ_n a square lattice region where there are n vertices of degree 4 on each row and each column. Λ_n is in *periodic boundary condition* if it forms a two-dimensional torus; the *free boundary condition* can be formulated in the following way: there are $n + 2$ vertices on each row and each column, where the "boundary vertices" are of degree 1 and don't need to satisfy the ice-rule (and don't take weights) in a valid six-vertex configuration. For convenience, we assume there are "virtual edges" connecting every two boundary vertices with unit distance on $\mathbb{Z}^2$. A virtual edge does not have orientations, serving only the purpose that every unit square inside the $(n + 1) \times (n + 1)$ region is closed.

Let Ω be the set of all valid configurations of the six-vertex model (Eulerian orientations) on Λ_n. The Glauber-dynamics Markov chain, which we will denote by $\mathcal{M}_G$, has state space Ω. To move from one configuration to another, this chain selects a unit square (a *face*) s on Λ_n (together with the virtual edges) uniformly at random. If all the non-virtual edges along the unit square s are oriented consistently (clockwise or counter-clockwise), the chain picks a direction d (clockwise or counter-clockwise) and reorients the non-virtual edges along s according to the Gibbs measure.

One can easily check that such transitions take valid configurations to valid configurations. Actually, this Markov chain is equivalent to that in [11] for sampling three-colorings on the faces of Λ_n. The ergodicity of that chain translates straightforwardly to the ergodicity of $\mathcal{M}_G$ (with free boundary conditions) thanks to the equivalence between Eulerian orientations and three-colorings on $\mathbb{Z}^2$. Besides, the heat-bath move indicates that the stationary distribution of $\mathcal{M}_G$ is the Gibbs distribution for the six-vertex model.

2.1.2 Directed-loop algorithm

The directed-loop algorithm Markov chain, denoted by $\mathcal{M}_D$, is formally defined in [7] for general 4-regular graphs, so here we only describe $\mathcal{M}_D$ at a high level.

The state space of $\mathcal{M}_D$ is not only Ω, the "perfect" Eulerian orientations, but also the set of all "near-perfect" Eulerian orientations, denoted by Ω'. For example, in Figure 3 the state τ_{ur} is in Ω and all other five states are in Ω'. We think of each edge in Λ_n as the two half-edges cut in the middle, and each of the half edge can be oriented independently. We say an orientation of all the half-edges is *perfect* (in Ω) if every pair of half-edges is oriented consistently and the ice-rule is satisfied at every vertex (except for boundary vertices under free boundary conditions); an orientation is *near-perfect* (in Ω') if there are exactly two pairs of half-edges p_1 and p_2 not oriented consistently and the ice-rule is satisfied at every vertex (except for boundary vertices under free boundary conditions), with the restriction that if two half-edges in p_1 are oriented toward each other then in p_2 the two half-edges must be oriented against each other and vice versa.

The transitions in $\mathcal{M}_D$ are Metropolis moves among "neighboring" states. An Ω state τ and an Ω' state τ' are neighboring if τ' can be transformed from τ by picking two half-edges e_1, e_2 incident to a vertex v with one pointing inwards v and the other pointing outwards v (or two half-edges e_1, e_2 on the boundary with one pointing towards the boundary and the other pointing against the boundary), and reverse the direction of e_1 and e_2 together. For instance, in Figure 3 $\{\tau_{ur}, \tau_{1b}\}$ and $\{\tau_{ur}, \tau_{1c}\}$ are two pairs of neighboring states. An Ω' state τ_1' and another Ω' state τ_2' are neighboring if τ_2' can be transformed from τ_1' by "shifting" one pair of conflicting half-edges one step away, while fixing the other pair of conflicting half-edges. For example, in Figure 3 τ_{1c} and τ_2 are neighboring to each other. $\mathcal{M}_D$ can be proved to be ergodic and converges to the Gibbs measure on $\Omega \cup \Omega'$ with both free boundary conditions and periodic boundary conditions [7].

2.2 Mixing time

The *mixing time* t_{mix} measures the time required by a Markov chain to evolve to be close to its stationary distribution, in terms of *total variation distance*. (The definition of mixing time can be found in [13].) We say a Markov chain is torpid mixing if the mixing time is exponentially large in the input size. A common technique to bound the mixing time is via bounding conductance, defined by Jerrum and Sinclair [28].

Let π denote the stationary distribution of an *ergodic* and *time reversible* ($\pi(x)P(x,y) = \pi(y)P(y,x)$ for any $x, y \in \Omega$) Markov chain $\mathcal{M}$ on a finite state space Ω, with transition probabilities $P(x,y)$, $x, y \in \Omega$. The *conductance* of $\mathcal{M}$ is defined by

$$\Phi = \Phi(\mathcal{M}) = \min_{\substack{S \subset \Omega \\ 0 < \pi(S) \leq \frac{1}{2}}} \frac{Q(S, \overline{S})}{\pi(S)},$$

where $Q(S, \bar{S})$ denotes the sum of $Q(x,y) = \pi(x)P(x,y)$ over edges in the transition graph of $\mathcal{M}$ with $x \in S$, and $y \in \overline{S} = \Omega \setminus S$.

In order to show a Markov chain mixes torpidly, we only need to prove that the conductance is (inverse) exponentially small due to the following bound [13]:

$$t_{\text{mix}} = t_{\text{mix}}\left(\frac{1}{4}\right) \geq \frac{1}{4\Phi}.$$

As is usually assumed, Markov chains studied in this paper are all *lazy* ($P(x,x) = \frac{1}{2}$ for any $x \in \Omega$) and transition probabilities ($P(x,y)$ for $x, y \in \Omega$) between neighboring states (where $P(x,y) > 0$) are at least inverse polynomially large. Therefore, armed with the above bound, we can prove the torpid mixing of a Markov chain if we can establish the following:

1. Partition the state space Ω into three subsets $\Omega_{\text{LEFT}} \cup \Omega_{\text{MIDDLE}} \cup \Omega_{\text{RIGHT}}$ as a disjoint union.
2. Show that for any state $\tau_l \in \Omega_{\text{LEFT}}$ and $\tau_r \in \Omega_{\text{RIGHT}}$, $P(\tau_l, \tau_r) = 0$. Under the assumption that the Markov chain is irreducible (i.e., the transition graph is strongly connected), this indicates that in order to go from states in Ω_{LEFT} to states in Ω_{RIGHT}, the Markov process has to go through the "middle states" Ω_{MIDDLE}.
3. Demonstrate that $\pi(\Omega_{\text{MIDDLE}})$ is exponentially small (compared with $\min(\pi(\Omega_{\text{LEFT}}), \pi(\Omega_{\text{RIGHT}}))$) in the input size. This means that starting from any state in Ω_{LEFT}, the probability of going through Ω_{MIDDLE} (and consequently to any state in Ω_{RIGHT} and reach stationarity) is exponentially small. Hence the conclusion of torpid mixing.

3 Ferroelectric phase

In this section we prove Theorem 1.1 that $\mathcal{M}_D$ in the directed-loop algorithm for the six-vertex model in the ferroelectric phase is torpid mixing on Λ_n with periodic boundary conditions.

For any parameter setting (a, b, c) in the ferroelectric phase, either $a > b + c$ or $b > a + c$. By symmetry, without loss of generality, suppose $a > b + c$. This implies that vertex configurations as shown in Figure 1-1 and Figure 1-2 have higher weights than others. Under the periodic boundary condition, there is a state τ_{ur} in which every vertical edge points upwards and every horizontal edge points to the right (Figure 3a), i.e., every vertex on Λ_n is in local configuration shown in Figure 1-1. The total weight of τ_{ur} is a^{n^2} as there are n^2 vertices on Λ_n.

For $\mathcal{M}_D$, the three-way partition of the state space $\Omega \cup \Omega'$ is as follows. Denote by T_i the states that can be reached from τ_{ur} in at most i steps of transitions where i is a nonnegative integer. Write $\partial T_i = T_i \setminus T_{i-1}$ for $i \geq 1$. Let $\Omega_{\text{LEFT}} = T_{n-1}$, $\Omega_{\text{MIDDLE}} = \partial T_n$, and $\Omega_{\text{RIGHT}} = (\Omega \cup \Omega') \setminus (\Omega_{\text{LEFT}} \cup \Omega_{\text{MIDDLE}})$. It is obvious that $\Omega \cup \Omega' = \Omega_{\text{LEFT}} \cup \Omega_{\text{MIDDLE}} \cup \Omega_{\text{RIGHT}}$ is a partition of the state space. Clearly $\tau_{ur} \in \Omega_{\text{LEFT}}$, thus the total weight of Ω_{LEFT} is no less than a^{n^2}, the weight of τ_{ur}.

Before proving the total weight of Ω_{MIDDLE} is exponentially small compared with that of Ω_{LEFT} or Ω_{RIGHT}, let us look at what is in T_i with $0 \leq i \leq n$. T_0 is just $\{\tau_{ur}\}$. ∂T_1 consists

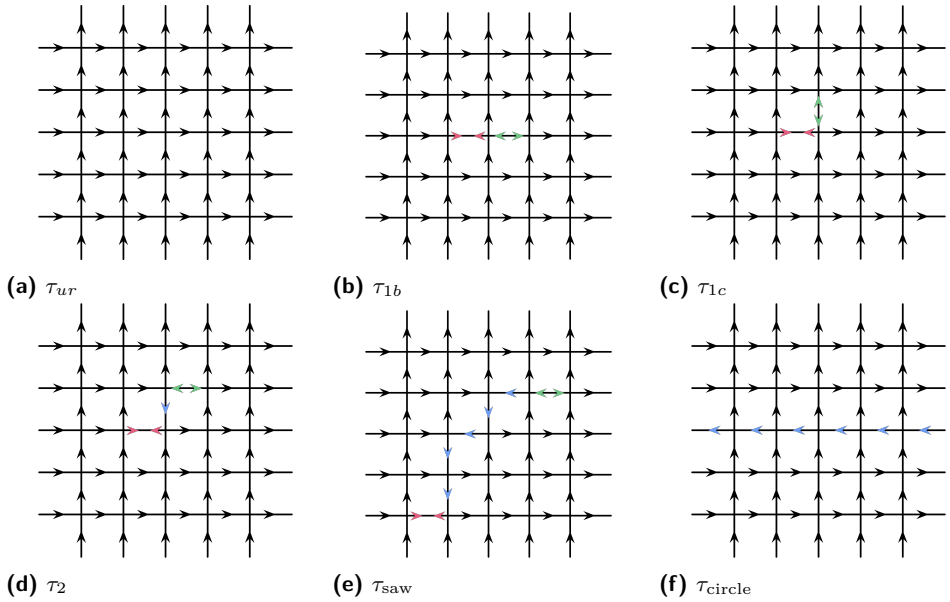


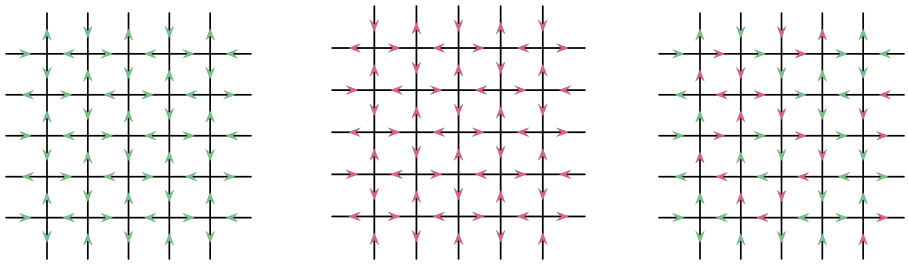
Figure 3 Some states in the state space of $\mathcal{M}_D$.

of all the states evolved from τ_{ur} by picking a vertex v on Λ_n and two incident half-edges (one pointing towards v and the other away from v), and then reversing the orientations on these two edges. After such a transition, two pairs of conflicting half-edges are created, so $\partial T_1 \subseteq \Omega'$.

For example, the states shown in Figure 3b (state τ_{1b}) and Figure 3c (state τ_{1c}) are in ∂T_1. The weight of τ_{1b} is $a^{n^2-1}b$ and that of τ_{1c} is $a^{n^2-1}c$. For every state in ∂T_1 obtained by transitions from τ_{ur}, there is exactly one vertex v^* on Λ_n no longer in the local configuration Figure 1-1. (Of course no vertex can be in state Figure 1-2.) Actually, depending on whether the two pairs of conflicting half-edges are: (1) both vertical, (2) both horizontal, or (3) one horizontal and the other vertical, the vertex v^* is in configuration shown in (1) Figure 1-3, (2) Figure 1-4, or (3) Figure 1-5/6, respectively. Therefore, every state in ∂T_1 has weight $a^{n^2-1}b$ in case (1) and case (2) or $a^{n^2-1}c$ in case (3).

Transitions from states in ∂T_1 to states in ∂T_2 are composed of "shifting" one of the two conflicting pairs of half-edges to a neighboring edge on Λ_n. For example, the state in Figure 3d is in ∂T_2. This process will result in exactly two vertices on Λ_n not in local configuration Figure 1-1 (nor in Figure 1-2). As a consequence, the weight of any state in ∂T_2 is among $a^{n^2-2}b^2$, $a^{n^2-2}bc$, and $a^{n^2-2}c^2$. The state shown in Figure 3d has weight $a^{n^2-2}c^2$.

This line of argument can be extended for ∂T_i for $1 \le i \le n$, in any state of which there are exactly i vertices on Λ_n not in local configuration Figure 1-1 (nor in Figure 1-2). When two conflicting pairs of half-edges are created in ∂T_1, one of them is above or to the right of another (or both). Denote the former by p_{ur} (the green pair in Figure 3) and the latter by p_{dl} (the red pair in Figure 3). Observe that as the Markov chain evolves, by a single step, from a state in ∂T_i to another in ∂T_{i+1} (where $1 \le i \le n-1$), either p_{ur} is "pushed" up or to the right, or p_{dl} down or to the left. For example, from Figure 3c to Figure 3d, p_{ur} is pushed to the right. By induction, p_{ur} is always above or to the right of p_{dl} (when $i < n$). A direct consequence is that there can be no state containing a closed circuit formed by the reversed edges (with regard to τ_{ur}) in ∂T_i until $i = n$. Therefore, the edges reversed in any state in ∂T_i ($1 \le i \le n$) can be seen as either a *self-avoiding walk* between the middle points of the



(a) τ_G **(b)** τ_R **(c)** τ

Figure 4 Some states in the state space of $\mathcal{M}_G$.

two pairs of conflicting half-edges (e.g. Figure 3e) or a *self-avoiding circuit* (e.g. Figure 3f). In fact, when the reversed edges form a circuit, the circuit must "go straightforward" at each step. This circuit is a circle parallel or perpendicular to the torus equatorial plane. The weight of any state in ∂T_i is $a^{n^2-i}b^j c^k$ with $i = j + k$, where the values of j and k depend on how many "turnarounds" are there in the self-avoiding walk.

Therefore, the total weight of states in ∂T_n is at most $n^2 \cdot a^{n^2-n}(b+c)^n$, where n^2 is an upper bound on all the possible starting points for self-avoiding walks, and each monomial in $(b+c)^n$ is from a unique self-avoiding walk. Combining with the fact that total weight of Ω_{LEFT} is at least a^{n^2} (the weight of τ_{ur}) and is at most that of Ω_{RIGHT} (because there is a weight-preserving injective map from Ω_{LEFT} to Ω_{RIGHT} by reversing orientations of all the edges), we know that the conductance of $\mathcal{M}_D$ is at most $\frac{n^2 \cdot a^{n^2-n}(b+c)^n}{a^{n^2}} = n^2 \left(\frac{b+c}{a}\right)^n$. This is exponentially small in n since $a > b + c$ are fixed constants in the ferroelectric phase.

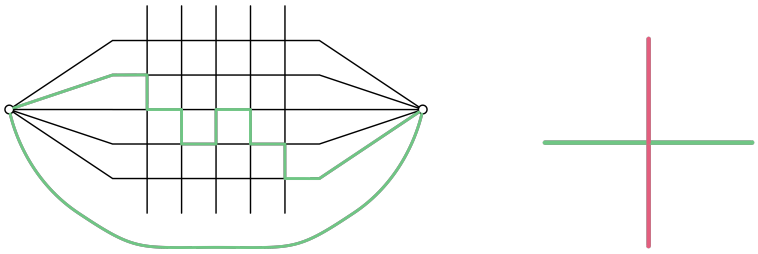
4 Anti-ferroelectric phase

In this section, we prove the following theorem which is part of Theorem 1.2. After proving Theorem 4.1, we state the ideas needed to extend it to Theorem 1.2, the full proof of which is omitted due to space limit.

As we did in the ferroelectric phase, the intuition behind our proof for the anti-ferroelectric phase is to find a partition $\Omega = \Omega_{\text{LEFT}} \cup \Omega_{\text{MIDDLE}} \cup \Omega_{\text{RIGHT}}$ of the state space of $\mathcal{M}_G$, i.e., all the Eulerian orientations on Λ_n. However, the strategy is different from that used in Section 3 — here the subset Ω_{MIDDLE} is determined in terms of a topological obstruction.

▶ **Theorem 4.1** (Anti-ferroelectric phase). *Glauber dynamics for the six-vertex model under parameter settings (a, b, c) with* $c \geq 2.639 \max(a, b)$ *mix torpidly on* Λ_n *with free boundary conditions.*

Observe that there are two states in Ω with maximum weights: τ_G (Figure 4a) and τ_R (Figure 4b) where every vertex is in local configuration Figure 1-5 or Figure 1-6, and thus has vertex weight c. Since τ_G and τ_R are total reversals of each other in edge orientations, for any edge in any state $\tau \in \Omega$, it is oriented either as in τ_G or as in τ_R. Let us call an edge to be *green* if it is oriented as is in τ_G and *red* otherwise. Observe that in order to satisfy the ice-rule (2-in-2-out), the number of green (and thus also two red) edges incident to any vertex (except for the boundary vertices) is always even $(0, 2,$ or $4)$, and if there are two green (or red) edges they must be *rotationally adjacent* to each other. See Figure 4c for an example. Also note that the four edges along a unit square on $\mathbb{Z}^2$ are all red edges or all green edges if and only if they are oriented consistently, hence flippable by a single move of $\mathcal{M}_G$.



(a) A closed curve $\mathcal{C}$. **(b)** An impossible configuration.

Figure 5

We say a simple path from a horizontal edge on the left boundary of Λ_n to a horizontal edge on the right boundary of Λ_n is a *horizontal green (or red) bridge* if the path consists of only green (or red, respectively) edges; a *vertical green (or red) bridge* is defined similarly. A state $\tau \in \Omega$ has a *green cross* if it has both a green horizontal bridge and a green vertical bridge; a *red cross* is defined similarly. Let $C_{\mathrm{G}} \subset \Omega$ denote the states having a green cross and C_{R} the states having a red cross. In the following lemma, we prove that $C_{\mathrm{G}} \cap C_{\mathrm{R}} = \emptyset$.

▶ **Lemma 4.2.** *A green cross and a red cross cannot coexist.*

Proof. It suffices to show that a green horizontal bridge precludes a red vertical bridge. Consider a virtual point v_{L} sitting to the left of Λ_n connected by an edge to every (external) vertices of Λ_n on the left boundary, and another virtual point v_{R} connected by an edge to every vertex of Λ_n on the right boundary. Connect v_{L} and v_{R} by an edge below Λ_n.

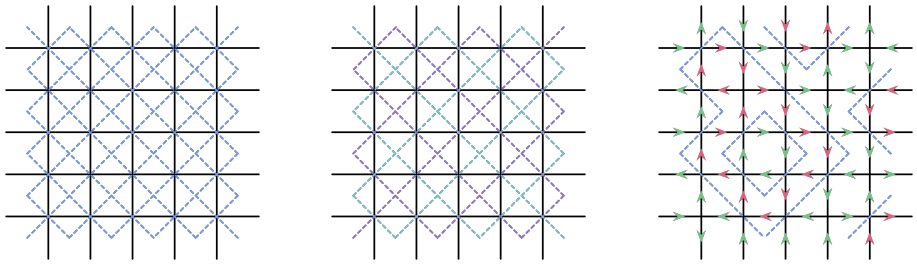
If there is a green horizontal bridge, then by definition there is a continuous closed curve $\mathcal{C}$ formed by the bridge and some edges we added (Figure 5a). According to the *Jordan Curve Theorem*, $\mathcal{C}$ separates the plane into two disjoint regions, the inside and the outside. Vertices of Λ_n that are on the bottom boundary are inside; vertices on the top boundary are outside. Therefore, in order to have a red vertical bridge, there must be a simple red path going across $\mathcal{C}$. That is to say, a red vertical bridge must cross the green horizontal bridge.

However, this is impossible. Clearly, being of different colors, a red bridge and a green bridge cannot share any edge. Since the local configuration shown in Figure 5b means that the four edges incident to a vertex (i, j) on Λ_n are all pointing inwards (when $i + j$ is even) or all pointing outwards (when $i + j$ is odd), it is not allowed in any valid states of six-vertex configurations. Similarly, the local configuration of a vertex surrounded by two red horizontal edges and two green vertical edges (a 90 degree rotation of Figure 5b) is also not allowed. ◀

Next we characterize the states in $\Omega \setminus (C_{\mathrm{G}} \cup C_{\mathrm{R}})$. Define a shifted lattice[1] L to be $\mathbb{Z}^2 + \left(\frac{1}{2}, \frac{1}{2}\right)$ where two points (a, b) and (c, d) in L are neighbors if $|a - c| = |b - d| = 1$ (a, b, c, and d are all half integers), i.e., they are at the center of a square in $\mathbb{Z}^2$ and are connected by "diagonal" edges of length $\sqrt{2}$. An example of L and its relationship with $\mathbb{Z}^2$ is shown in Figure 6a. L is not connected — it is composed of two sub-lattices L_0 and L_1 (depicted with different colors in Figure 6b). Denote by L_n the restriction of L on the finite region inside Λ_n. Note that in graph theoretical terms, the square lattice Λ_n is planar and 4-regular, and thus can be seen as the *medial graph* of two planar graphs. In fact, they are L_0 and L_1 (restricted onto L_n).

From now on, we use Λ_n-vertices/edges as an abbreviation for vertices/edges in Λ_n; and we use L_n-vertices/edges and other similar notations whenever it has a clear meaning in

[1] Strictly speaking, a lattice is a discrete subgroup of $\mathbb{R}^n$. A shifted copy of a lattice does not contain 0.



(a) L **(b)** L_0 and L_1 **(c)** τ and L_τ

Figure 6

the context. For any state $\tau \in \Omega$, there is a subset L_τ of L_n-edges associated with τ. Each L_n-edge e "goes diagonally through" exactly one Λ_n-vertex, denoted by v_e. We say e is in L_τ if and only if the four Λ_n-edges incident to v_e are 2-green-2-red and e *separates* the two green edges from the two red edges (Remember that in this case edges in the same color must be rotationally adjacent to each other). See Figure 6c for an instance of a state τ and its associated L_τ. In the following we abuse the notation and use L_τ as its induced subgraph of L. This view was adopted by [6] for establishing the existence of the spontaneous staggered polarization in the anti-ferroelectric phase of the six-vertex model.

For any $\tau \in \Omega$ and L_τ, we make the following observations:

- There is always an even number of L_τ-edges meeting at any L_n-vertex, except for the L_n-vertices on the boundary. Because this number is equal to the number of times for the color change on the four Λ_n-edges surrounding the L_n-vertex, if we start from any one of the four Λ_n-edges and go rotationally over the four Λ_n-edges, which is even.
- For any Λ_n vertex, there can be at most one L_τ-edge going through which is either in L_0 or in L_1.
- If $\bar{\tau} \in \Omega$ is the state by a total edge reversal of τ, then $L_{\bar{\tau}} = L_\tau$.

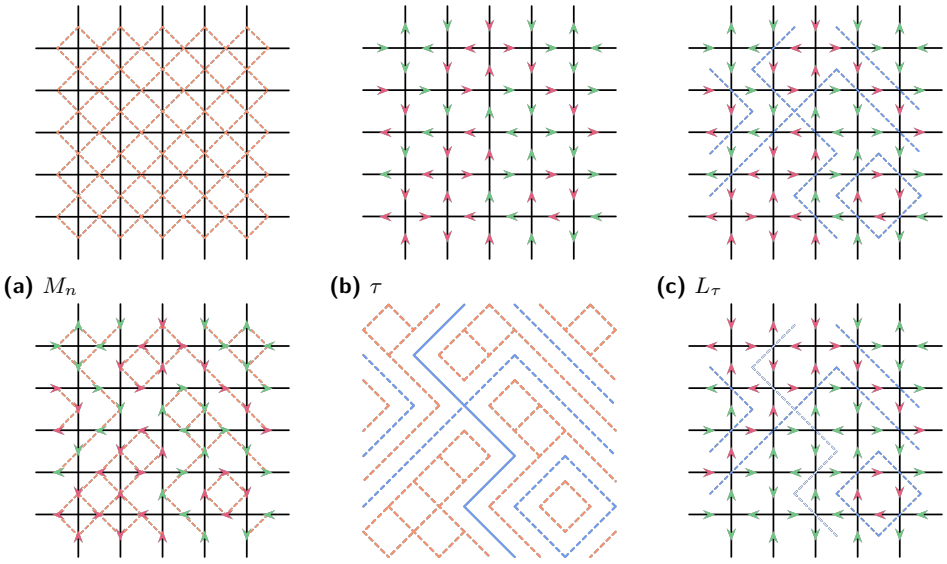
For a state $\tau \in \Omega$, we say τ has a *horizontal (or vertical) fault line* if there is a self-avoiding path in L_τ connecting a L_n-vertex on the left (top, respectively) boundary of L_n to a L_n-vertex on the right (bottom, respectively) boundary of L_n. See Figure 7c for an example where a state has both a horizontal fault line and a vertical fault line. Denote by C_{FL} the set of states containing a horizontal fault line or a vertical fault line. Since a fault line separates green edges from red edges, a vertical (horizontal) fault line precludes any horizontal (vertical, respectively) monochromatic bridge (the proof is basically the same as Lemma 4.2). This is to say, C_{G}, C_{FL}, and C_{R} are pairwise disjoint. Next we show the following lemma and its direct implication (Corollary 4.4).

▶ **Lemma 4.3.** *If in a state τ there is no monochromatic cross, then there is a fault line.*

Proof. If there is no monochromatic cross (i.e., a green cross or a red cross) in τ, we can assume that

there is no green horizontal bridge and there is no red horizontal bridge. $(*)$

- Suppose τ has a green horizontal bridge. There is no red vertical bridge since it cannot "cross" the green horizontal bridge; there is no green vertical bridge since there is no green cross. Therefore, this case is symmetric to $(*)$, switching horizontal for vertical.
- Suppose τ has a red horizontal bridge. This case is similar to the above case, and thus is also symmetric to $(*)$.



(a) M_n **(b)** τ **(c)** L_τ

(d) M_τ **(e)** Duality **(f)** The injective map

Figure 7

Next we show there is a vertical fault line if there is no monochromatic horizontal bridge. We introduce another graph M_n that is the *medial graph* of Λ_n where every vertex of M_n corresponds to an edge of Λ_n, i.e., two M_n-vertices are neighboring if the two corresponding Λ_n-edges are rotationally adjacent. Note that M_n is part of another shifted square lattice. An example of M_n and its relationship with $\mathbb{Z}^2$ is shown in Figure 7a. For any state $\tau \in \Omega$, there is a subset M_τ of M_n-edges associated with τ. An M_n-edge e is in M_τ if the two vertices that e is incident to (as Λ_n-edges) have the same color (both green or both red). See Figure 7d for an example.

Observe that the correspondence between Λ_n-edges and M_n-vertices translates into the correspondence between simple monochromatic paths in Λ_n to simple connected paths in M_τ. In fact, the connected components in M_τ, capturing the notion of monochromatic regions of Λ_n-edges, and connected components in L_τ, capturing the notion of separation between regions of Λ_n-edges of different colors, are in a *dual* relationship.

This duality is depicted in Figure 7e and helps us find a fault line. Let V_M be the collection of M_n-vertices that can be reached from the left boundary of M_n by a simple path in M_τ. Since there is no monochromatic horizontal bridge, V_M does not contain any M_n-vertex on the right boundary. As a consequence, there is a *cutset* in M_n separating V_M from the right boundary. This cutset, composed of M_n-edges, corresponds to a vertical fault line. For instance, in Figure 7e the blue solid L_τ-path γ is a fault line defined by the above argument. ◀

▶ **Corollary 4.4.** *$\Omega = C_G \cup C_{FL} \cup C_R$ is a partition of the state space.*

Before moving on to prove Theorem 4.1, we introduce the notion of *almost fault lines*. A *horizontal (or vertical) almost fault line* is a self-avoiding L_n-path connecting a L_n-vertex on the left boundary of L_n to a L_n-vertex on the right boundary of L_n where all edges except for one are in L_τ. Denote by C_{AFL} the set of states containing an almost fault line. Let ∂C_{G} be the set of states outside C_{G} which are one-flip away from C_{G} in the state space of $\mathcal{M}_G$.

▶ **Lemma 4.5.** *$\partial C_G \subset C_{FL} \cup C_{AFL}$.*

(a) τ_g **(b)** τ_∂

Figure 8 A step in $\mathcal{M}_G$.

Proof. If a state $\tau_\partial \in \partial C_{\mathrm{G}}$ is not in C_{FL} (does not have a fault line), then by Corollary 4.4 $\tau_\partial \in C_{\mathrm{R}}$ since by definition it is outside of C_{G}. Because τ_∂ is one move away from C_{G}, there exists a state $\tau_g \in C_{\mathrm{G}}$ such that flipping four monochromatic edges along a unit square s on $\mathbb{Z}^2$ yields τ_∂. We know that in τ_g there exists a green cross and no red cross; in τ_∂ there exists a red cross and no green cross. Then it must be true that the four edges along s are all green in τ_g and all red in τ_∂. See Figure 8 for a pictorial illustration. Moreover, any green cross in τ_g must contain edges along s and so is any red cross in τ_∂; otherwise, a green horizontal (vertical) bridge in τ_g must go across a red vertical (or horizontal, respectively) bridge in τ_∂ at some other place on Λ_n, which is impossible.

Therefore, there exists a simple green path Γ_{G} from some vertices on s to the top boundary of Λ_n and a simple red path Γ_{R} from some vertices on s to the top boundary of Λ_n. In the following, we prove that the above conditions suffice to show that there exists an L_τ-path from s to the top boundary of L_n. Similar conclusions can be made for the existence of L_τ-paths from s to the bottom boundary of L_n. By adding at most one L_n-edge, we can concatenate these paths to obtain a vertical almost fault line.

Γ_{G} and Γ_{R} cannot cross each other (Figure 5). Without loss of generality, suppose Γ_{G} is to the left of Γ_{R}. Then we use the medial lattice view M_{τ_∂} as in the proof of Lemma 4.3. Γ_{G} corresponds to a connected component m_{G} in M_{τ_∂}. Denote by V_{G} the set of M_n-vertices which can be reached by m_{G} in M_{τ_∂}, and let V'_{G} be V_{G} together with the set of M_n-vertices which are separated from the right boundary of M_n by V_{G}. Then there is a *cutset* in M_n separating V'_{G} from the right boundary. This cutset, composed of M_n-edges, corresponds to a dual L_τ-path from s to the top boundary of L_n. ◀

Now we are ready to show Theorem 4.1. Let $\Omega_{\mathrm{LEFT}} = C_{\mathrm{G}}$, $\Omega_{\mathrm{MIDDLE}} = C_{\mathrm{FL}} \cup (C_{\mathrm{AFL}} \cap C_{\mathrm{R}})$, and $\Omega_{\mathrm{RIGHT}} = C_{\mathrm{R}} \setminus C_{\mathrm{AFL}}$. Theorem 4.1 is a consequence of the following lemma which uses a *Peierls argument* to show that $\pi(C_{\mathrm{FL}} \cup C_{\mathrm{AFL}})$ is exponentially small.

▶ **Lemma 4.6.** $\pi(C_{FL} \cup C_{AFL}) \leq O(n) \left(\frac{2.639 \max(a,b)}{c}\right)^n$.

Proof. For a self-avoiding path γ in L_n connecting a vertex on the top boundary to a vertex on the bottom boundary, denote by F_γ the set of states in Ω that contain γ as vertical fault line or almost fault line. Reversing directions of all the edges to the left side of γ defines an injective mapping from F_γ to $\Omega \setminus F_\gamma$ that magnifies probability by a factor of at least $\frac{\min(a,b)}{c} \cdot \left(\frac{c}{\max(a,b)}\right)^{|\gamma|-1}$. This is because: if γ is a fault line in a state, every Λ_n-vertex sitting on γ would have four incident edges in the same color after the map, which increase its weight to c (from a or b); if γ is an almost fault line in a state, the above is true except that for one Λ_n-vertex, its weight decrease from c to a or b after the map, as orientations on half of the four monochromatic edges are reversed. This indicates that $\pi(F_\gamma) \leq \frac{c}{\min(a,b)} \cdot \left(\frac{\max(a,b)}{c}\right)^{|\gamma|-1}$.

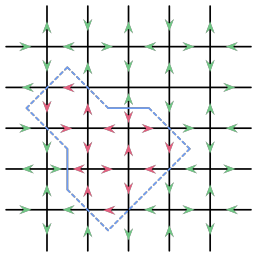


Figure 9 A state in $\mathcal{M}_D$.

See Figure 7f for an example. The same goes for horizontal (almost) fault lines.

Since every (almost) fault line is a self-avoiding walk on L, the number of fault lines of length l is upper bounded by $2n$ times the number of self-avoiding walks of that length starting at a vertex on the left or bottom boundary. The latter can be bounded by an well-studied estimate μ^l on the number of self-avoiding walks of length l on $\mathbb{Z}^2$, where μ is called the *connective constant*. The best proved bound is $\mu \approx 2.638158\cdots$ [12]. Summing this over fault lines of length from n to n^2 completes the proof. ◀

We have proved the torpid mixing of Glauber dynamics for the six-vertex model on the lattice region Λ_n with free boundary conditions. Next we state the idea to extend the proof for the case when the Markov chain is $\mathcal{M}_D$ and the case when the boundary of $\widetilde{\Lambda}_n$ is periodic. Theorem 1.2 is a combination of Theorem 4.1 and the extensions.

To extend Theorem 4.1 to hold for the directed-loop algorithm $\mathcal{M}_{\mathcal{D}}$ whose state space is $\Omega \cup \Omega'$, we need to pay extra attention for the states in Ω', the near-perfect Eulerian orientations. For a state $\tau' \in \Omega'$, there are two "defects" on the edges (Figure 9). Apart from the diagonal L_n-edges, there are two $\left(\mathbb{Z}^2 + \left(\frac{1}{2}, \frac{1}{2}\right)\right)$-edges separating green (half-)edges from red (half-)edges. The adaption we make is to put such $\left(\mathbb{Z}^2 + \left(\frac{1}{2}, \frac{1}{2}\right)\right)$-edges also into the set $L_{\tau'}$. Notice that a connected component in $L_{\tau'}$ could possibly lie on L_0-edges as well as L_1-edges. Everything we prove is still correct if we allow fault lines to have $\left(\mathbb{Z}^2 + \left(\frac{1}{2}, \frac{1}{2}\right)\right)$-edges. Due to the possible positions of such two defects, the weight of $C_{\mathrm{FL}} \cup C_{\mathrm{AFL}}$ only increases by a polynomial factor in n, hence not affecting the fact of torpid mixing.

To extend Theorem 4.1 to hold for $\widetilde{\Lambda}_n$ with periodic boundary condition (i.e., a 2-dimensional torus), we make the following modification. When n is even, there still are two states with maximum weights c^{n^2} (similar to τ_G and τ_R in Figure 4). Again, for any state τ, $\widetilde{\Lambda}_n$-edges can be classified as green or red, and its associated L_τ separates $\widetilde{\Lambda}_n$-edges of different colors. For the 2-dimensional torus T^2, the homology group $H_1(T^2) \cong \mathbb{Z} \times \mathbb{Z}$. We say a state τ has a *green (or red) cross* if there are two *non-contractable* cycles of green (or red, respectively) edges of *homology classes* (a_1, b_1) and (a_2, b_2) with $\det \left[\begin{smallmatrix} a_1 a_2 \\ b_1 b_2 \end{smallmatrix}\right] \neq 0$; τ has *a pair of* fault lines if there is a pair of non-contractable cycles of L_τ-edges. (By parity, if there is one L_τ-cycle there must be two.) Then the proofs in this section can be naturally adapted for the torus case, and the torpid mixing result follows.

References

1 David Allison and Nicolai Reshetikhin. Numerical study of the 6-vertex model with domain wall boundary conditions. *Annales de l'Institut Fourier*, 55(6):1847–1869, 2005. URL: `http://eudml.org/doc/116236`.

2 G. T. Barkema and M. E. J. Newman. Monte Carlo simulation of ice models. *Phys. Rev. E*, 57:1155–1166, Jan 1998. `doi:10.1103/PhysRevE.57.1155`.

3 R. J. Baxter. *Exactly Solved Models in Statistical Mechanics*. Academic Press, 1982. doi:10.1007/978-3-642-73193-8_19.

4 Antonio Blanca, David Galvin, Dana Randall, and Prasad Tetali. Phase coexistence and slow mixing for the hard-core model on $\mathbb{Z}^2$. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, pages 379–394, 2013. doi:10.1007/978-3-642-40328-6_27.

5 Christian Borgs, Jennifer T. Chayes, Jeong Han Kim, Alan Frieze, Prasad Tetali, Eric Vigoda, and Van Ha Vu. Torpid mixing of some Monte Carlo Markov chain algorithms in statistical physics. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 218–229, 1999. URL: http://dl.acm.org/citation.cfm?id=795665.796518.

6 H. J. Brascamp, H. Kunz, and F. Y. Wu. Some rigorous results for the vertex model in statistical mechanics. *Journal of Mathematical Physics*, 14(12):1927–1932, 1973. doi:10.1063/1.1666271.

7 Jin-Yi Cai, Tianyu Liu, and Pinyan Lu. Approximability of the six-vertex model. *CoRR*, abs/1712.05880, 2017. arXiv:1712.05880.

8 F. Cesi, G. Guadagni, F. Martinelli, and R. H. Schonmann. On the two-dimensional stochastic Ising model in the phase coexistence region near the critical point. *Journal of Statistical Physics*, 85(1):55–102, Oct 1996. doi:10.1007/BF02175556.

9 K. Eloranta. Diamond Ice. *Journal of Statistical Physics*, 96:1091–1109, 1999. doi:10.1023/A:1004644418182.

10 Chungpeng Fan and F. Y. Wu. General lattice model of phase transitions. *Phys. Rev. B*, 2:723–733, Aug 1970. doi:10.1103/PhysRevB.2.723.

11 Leslie Ann Goldberg, Russell Martin, and Mike Paterson. Random sampling of 3-colorings in $\mathbb{Z}^2$. *Random Structures & Algorithms*, 24(3):279–302, 2004. doi:10.1002/rsa.20002.

12 A.J. Guttmann and A.R. Conway. Square lattice self-avoiding walks and polygons. *Annals of Combinatorics*, 5(3):319–345, Dec 2001. doi:10.1007/PL00013842.

13 David A. Levin, Yuval Peres, and Elizabeth L. Wilmer. *Markov chains and mixing times*. American Mathematical Society, 2006.

14 Elliott H. Lieb. Exact solution of the F model of an antiferroelectric. *Phys. Rev. Lett.*, 18:1046–1048, Jun 1967. doi:10.1103/PhysRevLett.18.1046.

15 Elliott H. Lieb. Exact solution of the two-dimensional Slater KDP model of a ferroelectric. *Phys. Rev. Lett.*, 19:108–110, Jul 1967. doi:10.1103/PhysRevLett.19.108.

16 Elliott H. Lieb. Residual entropy of square ice. *Phys. Rev.*, 162:162–172, Oct 1967. doi:10.1103/PhysRev.162.162.

17 Eyal Lubetzky and Allan Sly. Critical Ising on the square lattice mixes in polynomial time. *Communications in Mathematical Physics*, 313(3):815–836, Aug 2012. doi:10.1007/s00220-012-1460-9.

18 Michael Luby, Dana Randall, and Alistair Sinclair. Markov chain algorithms for planar lattice structures. *SIAM Journal on Computing*, 31(1):167–192, 2001. doi:10.1137/S0097539799360355.

19 I Lyberg, V Korepin, and J Viti. The density profile of the six vertex model with domain wall boundary conditions. *Journal of Statistical Mechanics: Theory and Experiment*, 2017(5):053103, 2017. URL: http://stacks.iop.org/1742-5468/2017/i=5/a=053103.

20 F. Martinelli and E. Olivieri. Approach to equilibrium of Glauber dynamics in the one phase region. I. the attractive case. *Communications in Mathematical Physics*, 161(3):447–486, Apr 1994. doi:10.1007/BF02101929.

21 F. Martinelli and E. Olivieri. Approach to equilibrium of Glauber dynamics in the one phase region. II. the general case. *Communications in Mathematical Physics*, 161(3):487–514, Apr 1994. doi:10.1007/BF02101930.

22 M. Mihail and P. Winkler. On the number of Eulerian orientations of a graph. *Algorithmica*, 16(4):402–414, Oct 1996. doi:10.1007/BF01940872.

23 Linus Pauling. The structure and entropy of ice and of other crystals with some randomness of atomic arrangement. *Journal of the American Chemical Society*, 57(12):2680–2684, 1935. doi:10.1021/ja01315a102.

24 R. Peierls. Statistical theory of adsorption with interaction between the adsorbed atoms. *Mathematical Proceedings of the Cambridge Philosophical Society*, 32(3):471–476, 1936. doi:10.1017/S0305004100019162.

25 Aneesur Rahman and Frank H. Stillinger. Proton distribution in ice and the Kirkwood correlation factor. *The Journal of Chemical Physics*, 57(9):4009–4017, 1972. doi:10.1063/1.1678874.

26 Dana Randall. Slow mixing of Glauber dynamics via topological obstructions. In *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithm (SODA)*, pages 870–879, 2006. URL: http://dl.acm.org/citation.cfm?id=1109557.1109653.

27 Dana Randall and Prasad Tetali. Analyzing Glauber dynamics by comparison of Markov chains. *Journal of Mathematical Physics*, 41(3):1598–1615, 2000. doi:10.1063/1.533199.

28 Alistair Sinclair and Mark Jerrum. Approximate counting, uniform generation and rapidly mixing Markov chains. *Information and Computation*, 82(1):93–133, 1989. doi:10.1016/0890-5401(89)90067-9.

29 Bill Sutherland. Exact solution of a two-dimensional model for hydrogen-bonded crystals. *Phys. Rev. Lett.*, 19:103–104, Jul 1967. doi:10.1103/PhysRevLett.19.103.

30 Olav F. Syljuåsen and M. B. Zvonarev. Directed-loop Monte Carlo simulations of vertex models. *Phys. Rev. E*, 70:016118, Jul 2004. doi:10.1103/PhysRevE.70.016118.

31 A. Yanagawa and J.F. Nagle. Calculations of correlation functions for two-dimensional square ice. *Chemical Physics*, 43(3):329–339, 1979. doi:10.1016/0301-0104(79)85201-5.

On the Testability of Graph Partition Properties

Yonatan Nakar[1]
Tel Aviv University, Tel Aviv, Israel
yonatannakar@mail.tau.ac.il

Dana Ron[1]
Tel Aviv University, Tel Aviv, Israel
danaron@tau.ac.il

Abstract

In this work we study the testability of a family of graph partition properties that generalizes a family previously studied by Goldreich, Goldwasser, and Ron (*Journal of the ACM, 1998*). While the family studied by Goldreich, Goldwasser, and Ron includes a variety of natural properties, such as k-colorability and containing a large cut, it does not include other properties of interest, such as split graphs, and more generally (p, q)-colorable graphs. The generalization we consider allows us to impose constraints on the edge-densities within and between parts (relative to the sizes of the parts). We denote the family studied in this work by $\mathcal{GPP}$.

We first show that all properties in $\mathcal{GPP}$ have a testing algorithm whose query complexity is polynomial in $1/\epsilon$, where ϵ is the given proximity parameter (and there is no dependence on the size of the graph). As the testing algorithm has two-sided error, we next address the question of which properties in $\mathcal{GPP}$ can be tested with one-sided error and query complexity polynomial in $1/\epsilon$. We answer this question by establishing a characterization result. Namely, we define a subfamily $\mathcal{GPP}_{0,1}$ of $\mathcal{GPP}$ and show that a property $P \in \mathcal{GPP}$ is testable by a one-sided error algorithm that has query complexity $\text{poly}(1/\epsilon)$ if and only if $P \in \mathcal{GPP}_{0,1}$.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis

Keywords and phrases Graph Partition Properties

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.53

Acknowledgements We thank the anonymous reviewers of this paper for their helpful comments.

1 Introduction

In graph property testing, the goal is to decide whether a graph satisfies a prespecified property P or is far from satisfying P. To this end, the testing algorithm is given query access to the adjacency matrix of the input graph so that the algorithm can check whether there is an edge between any given pair of vertices.[2] A graph G over n vertices is said to be ϵ-far from satisfying P if it is necessary to add or delete more than ϵn^2 edges in order to turn G into a graph satisfying P. A *tester* for a graph property P is a randomized algorithm, which given query access to the graph, distinguishes with high constant probability between the case where G satisfies P and the case where G is ϵ-far from satisfying P. The tester should make the distinction between the two cases by observing a very small portion of the input graph. In other words, the tester must have sublinear query complexity.

We focus on properties that can be tested with no dependence on n. In particular, the query complexity of the testers we consider depends only on the proximity parameter ϵ,

[1] This research was partially supported by the Israel Science Foundation grant No. 671/13.
[2] Here we refer to what is known as the Dense Graph Model or the Adjacency Matrix Model [12].

and the decisions of the testers do not depend on n as well. We call such graph properties *input-size oblivious testable*. Alon et al. [3] presented a complete characterization of input-size oblivious testable graph properties. Independently, Borgs et al.[9] obtained an analytic characterization of such properties through the theory of graph limits. However, while the query complexity of the tester emerging from the characterization of Alon et al. does not depend on the graph size, it could be super-polynomial in $\frac{1}{\epsilon}$. For example, the property of being triangle-free is input-size oblivious testable, but the query complexity of the best known tester for triangle-freeness is a tower function of $\frac{1}{\epsilon}$ [10]. Further, there exists a super-polynomial lower bound on the query complexity of testing triangle-freeness [1, 6]. Naturally, we strive to design testers with query complexity that is polynomial in $\frac{1}{\epsilon}$.

In this paper we consider a family of graph partition properties. This family of properties, which will be defined shortly, generalizes a family of graph partition properties that was introduced by Goldreich, Goldwasser, and Ron [12]. Examples of properties covered by their framework include bipartiteness, k-colorability, and the property of having a cut of at least βn^2 edges for some $\beta \in [0,1]$. Their framework, while fairly general, lacks an ingredient that is necessary for specifying many natural graph partition properties such as split graphs (or more generally (p,q)-colorable graphs), probe complete graphs, and bisplit graphs.[3]

Given a graph $G=(V,E)$, a partition $(V_1,...,V_k)$ of V, and a pair of parts V_i, V_j (possibly $i=j$) we denote by $e_G(V_i,V_j)$ the number of edges in G between the part V_i and part V_j (if $i=j$, then the notation refers to the number of edges within the part). Following the definitions in [12], the notation $e_G(V_i,V_j)$ counts each edge twice (both (u,v) and (v,u)) and when $i=j$ we also allow self-loops. That is, $e_G(V_i,V_j)$ counts the number of ones in the adjacency matrix representing the graph G. Also, we denote by $\overline{e}_G(V_i,V_j)$ the number of nonedges between the two parts.

Each property in the family of *Graph Partition Properties* considered in this work, is defined by an integer parameter k and $O(k^2)$ additional parameters in $[0,1]$. Informally, a graph has the property if its vertices can be partitioned into k subsets such that the sizes of the subsets and the number of edges between pairs of subsets and within the subsets obey the constraints defined by the parameters of the property. Formally, a Graph Partition Property P is parameterized by an integer k denoting the number of parts and by the following parameters in the interval $[0,1]$:

1. Bounds on each part's size: for each $1 \le i \le k$ we have ρ_i^L, ρ_i^U s.t. part V_i must satisfy $\rho_i^L n \le |V_i| \le \rho_i^U n$.
2. Absolute bounds on the number of edges within each part: for each $1 \le i \le k$ we have ρ_{ii}^L, ρ_{ii}^U s.t. part V_i must satisfy $\rho_{ii}^L n^2 \le e_G(V_i,V_i) \le \rho_{ii}^U n^2$.
3. Absolute bounds on the number of edges between each pair of parts: for each pair $1 \le i,j \le k$ we have ρ_{ij}^L, ρ_{ij}^U s.t. the pair of parts V_i, V_j must satisfy $\rho_{ij}^L n^2 \le e_G(V_i,V_j) \le \rho_{ij}^U n^2$.
4. Relative bounds on the number of edges within each part: for each $1 \le i \le k$ we have $\alpha_{ii}^L, \alpha_{ii}^U$ s.t. part V_i must satisfy $\alpha_{ii}^L |V_i|^2 \le e_G(V_i,V_i) \le \alpha_{ii}^U |V_i|^2$.
5. Relative bounds on the number of edges between each pair of parts: for each pair $1 \le i,j \le k$ we have $\alpha_{ij}^L, \alpha_{ij}^U$ s.t. the pair of parts V_i, V_j must satisfy $2\alpha_{ij}^L |V_i| \cdot |V_j| \le e_G(V_i,V_j) \le 2\alpha_{ij}^U |V_i| \cdot |V_j|$.

[3] A graph is a *split graph* if it can be partitioned into an independent set and a clique. A graph is (p,q)-*colorable* if it can be partitioned into p cliques and q independent sets. A graph is *probe-complete* if it can be partitioned into an independent set and a clique such that every vertex in the independent set is adjacent to every vertex in the clique. A graph is *bisplit* if it can be partitioned into an independent set and a bi-clique.

The original graph partition framework that was introduced in [12] includes only Items 1–3. The absence of relative edge bounds makes the original framework weaker than the general framework we consider in this paper. In particular, using the original framework, one cannot express the notion of parts being cliques or the notion of a pair of parts being fully connected to each other. More generally, our framework enhances the expressive power of the original framework by adding the notion of edge densities,[4] a notion that does not exist in the original framework. We denote the class of graph partition properties (as defined above) by $\mathcal{GPP}$. We denote the class of graph partition properties that have no relative bounds on the number of edges (the one introduced in [12]) by $\mathcal{GPP}_{NR}$ (NR stands for Non-Relative).

We say that a graph property is $\text{poly}\left(\frac{1}{\epsilon}\right)$-testable if it is input-size oblivious testable and the tester's query complexity is polynomial in $\frac{1}{\epsilon}$. All the properties in the class $\mathcal{GPP}_{NR}$ are $\text{poly}\left(\frac{1}{\epsilon}\right)$-testable [12]. In this work, we first show how to use the algorithm presented in [12] as a subroutine to devise a tester for all the partition properties covered by our generalized framework, thus obtaining the following theorem:

▶ **Theorem 1.** *Every property $P \in \mathcal{GPP}$ is $\text{poly}\left(\frac{1}{\epsilon}\right)$-testable.*

While the query complexity of the tester implied by 1 is a polynomial function of $\frac{1}{\epsilon}$ as desired, it has the disadvantage of having two-sided error (just like the algorithm described in [12]). A tester has one-sided error if, whenever a graph G satisfies P, the tester determines this with probability 1. Clearly, a one-sided error tester is preferable to a two-sided error tester because a one-sided error tester is capable of providing a witness demonstrating that the property is not satisfied by the input graph. Combining the two desired features of polynomial dependence on $\frac{1}{\epsilon}$ and having one-sided error leads to the definition of *easily testable graph properties* (as defined in e.g. [7, 4, 11]).

▶ **Definition 2.** *A graph property P is easily-testable if P is $\text{poly}\left(\frac{1}{\epsilon}\right)$-testable and the tester has one-sided error.*

An example of an easily testable graph partition property is the property of being k-colorable [12, 5]. In this paper we address the question of characterizing the easily testable graph partition properties. We show that every graph partition property belonging to a restricted subset of $\mathcal{GPP}$, which we denote by $\mathcal{GPP}_{0,1}$ (and formally define below), is easily testable, and every graph partition property $P \notin \mathcal{GPP}_{0,1}$ is not easily testable. That is, while Theorem 1 implies that every graph partition property P is $\text{poly}\left(\frac{1}{\epsilon}\right)$-testable, only those properties in $\mathcal{GPP}_{0,1}$ are $\text{poly}\left(\frac{1}{\epsilon}\right)$-testable with one-sided error. An analogous result was established for the class $\mathcal{GPP}_{NR}$ by Goldreich and Trevisan [13]. However, as the class $\mathcal{GPP}$ is more general, the class $\mathcal{GPP}_{0,1}$ contains properties that are not covered by the class $\mathcal{GPP}_{NR}$. We build on some techniques used in [13] to establish our characterization, but our characterization does not result from [13] and we rely on different ideas to arrive at it.

The class $\mathcal{GPP}_{0,1}$ is a subclass of $\mathcal{GPP}$ for which the following holds. For every property $P \in \mathcal{GPP}_{0,1}$, there are no absolute bounds on the number of edges between or within parts. If P has a constraint on the edge density between a pair of parts, or within a part, the constraint must be either that the edge density is exactly 0 or that the edge density is exactly 1. In addition, P does not constrain the sizes of the parts. Formally, P is parameterized by an integer k denoting the number of parts and by a function $d_P : [k] \times [k] \to \{0, 1, \perp\}$ denoting the relative edge density that P imposes on parts i and j:

[4] From now on, when using the term edge density, we refer to the fraction of edges between the parts (or within the part) relative to the number of vertex pairs between the parts (or within the part).

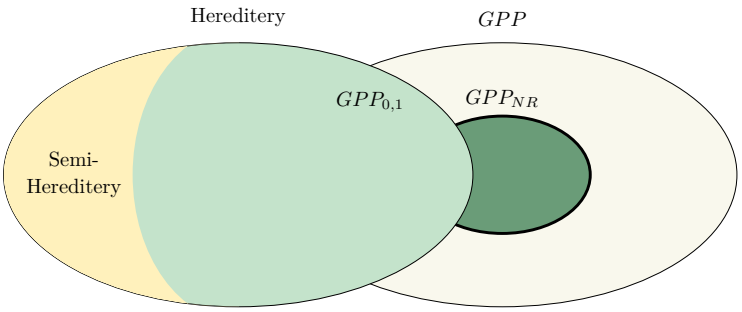


Figure 1 Inclusion relations among the graph partition properties.

$$d_P(i,j) = \begin{cases} 1 & \text{if every vertex in part } i \text{ should be connected to every vertex of part } j \\ 0 & \text{if there are no edges between part } i \text{ and part } j \\ \perp & \text{if any number of edges between part } i \text{ and } j \text{ is allowed} \end{cases}$$

Possibly, $i = j$ in which case it is the edge density within a single part. That is, if $d_P(i,i)$ is 0 or 1 then P forces part i to be an independent set or a clique respectively. It is clear from the definition that there are graph partition properties $P \in \mathcal{GPP}_{0,1}$ that are not part of the class $\mathcal{GPP}_{NR}$ (split graphs for instance).

The main result of our paper is a characterization of the easily testable graph partition properties.

▶ **Theorem 3.** *A graph property $P \in \mathcal{GPP}$ is easily testable if and only if $P \in \mathcal{GPP}_{0,1}$.*

Recall that a property is easily testable if it is testable by a one-sided error input-size oblivious tester whose query complexity is polynomial in $\frac{1}{\epsilon}$. If we remove the requirement that the dependence on $\frac{1}{\epsilon}$ is polynomial, then the property is said to be *strongly testable*. Alon and Shapira [8] define the notion of a property being *semi-hereditary* (which is a certain relaxation of being hereditary), and show that a graph property P is strongly testable if and only if P is semi-hereditary. Since the properties in $\mathcal{GPP}_{0,1}$ are clearly hereditary, and therefore semi-hereditary, the condition of Alon and Shapira implies that they are strongly testable. However, this is not enough to prove the "if" part of Theorem 3, because being strongly testable does not mean that the tester's query complexity is $\text{poly}\left(\frac{1}{\epsilon}\right)$. Therefore, to prove the "if" direction we give a $\text{poly}\left(\frac{1}{\epsilon}\right)$ one-sided error testing algorithm for the property. As for the "only if" direction, we could use [8] to get that if a property P in $\mathcal{GPP}$ is easily testable (and hence strongly testable), then it is semi-hereditary. We would then need to prove that if $P \in \mathcal{GPP}$ is semi-hereditary, then $P \in \mathcal{GPP}_{0,1}$. Establishing this claim would be essentially the same as our direct proof that if a property P in $\mathcal{GPP}$ is easily testable, then $P \in \mathcal{GPP}_{0,1}$, and would be based on the same proof ingredients.

We next give a brief summary of each of our results.

1.1 A Two-Sided Error Tester for properties in $\mathcal{GPP}$

In order to prove the existence of a (two-sided error) $\text{poly}\left(\frac{1}{\epsilon}\right)$-testing algorithm for $\mathcal{GPP}$ we show how to reduce the problem of testing properties in $\mathcal{GPP}$ to testing properties in $\mathcal{GPP}_{NR}$. Recall that the difference between properties in $\mathcal{GPP}$ and properties in $\mathcal{GPP}_{NR}$ is that the former include edge density constraints (that are relative to the sizes of the parts), while the latter include only absolute constraints on the sizes of the parts and the number of edges between/within them. We next give the high-level idea of the reduction.

Given a property $P \in \mathcal{GPP}$, we define a collection of properties in $\mathcal{GPP}_{NR}$, by *discretizing* P and replacing the edge-density constraints with absolute constraints on the number of edges. We then run the testing algorithm of [12], denoted $\mathcal{A}$, on G and each property in the constructed collection, with distance parameter $\frac{\epsilon}{2}$. If $\mathcal{A}$ accepts for at least one of these properties, then we accept, and otherwise we reject. The definition of the collection is such that if G satisfies P, then G satisfies at least one of the properties in the collection, so that our algorithm accepts with high constant probability. In order to show that if G is ϵ-far from P, then G is $\frac{\epsilon}{2}$-far from *every* property in the collection, we prove the contrapositive statement. That is, if for at least one of the properties P' in the collection, G is $\frac{\epsilon}{2}$-close to P', then G is $\frac{\epsilon}{2}$-close to P. While the first part of the analysis (regarding G that satisfies P) is fairly immediate, the second part (regarding G that is $\frac{\epsilon}{2}$-far from P) requires a more subtle analysis. Essentially, we need to show how to "fix" G (remove/add edges), so as to obtain a graph that satisfies P. This requires showing the existence of a partition $(V_1, \ldots, V_k)$ that obeys all the constraints defined by P, while closeness to P' only ensures the existence of a partition $(V'_1, \ldots, V'_k)$ that "almost" satisfies P'.

1.2 A One-Sided Error Tester for Properties in $\mathcal{GPP}_{0,1}$

The tester is very simple. It samples $\Theta\left(\frac{k\log(k)}{\epsilon^2}\right)$ vertices, checks whether or not the induced subgraph satisfies P and answers accordingly.[5] Since all the graph partition properties in $\mathcal{GPP}_{0,1}$ are hereditary, it clearly holds that if a graph G satisfies P, then every induced subgraph of G also does. Hence, if $G \in P$, the suggested tester accepts with a probability of 1.

The heart of the proof is in showing that if G is ϵ-far from satisfying P, where $P \in \mathcal{GPP}_{0,1}$, then with high constant probability, the subgraph induced by the sample does not satisfy P. In other words, we would like to show that with high constant probability over the choice of the sample S, every partition $(S_1, \ldots, S_k)$ of the sample violates at least one of the constraints defined by the property P. That is, there is a pair (u, v), where $u \in S_i$ and $v \in S_j$ such that either $(u, v) \in E$ while $d_P(i, j) = 0$, or $(u, v) \notin E$ while $d_P(i, j) = 1$. Such a partition is said to be *invalid*. In order to prove this claim we extend the analysis of Alon and Krivelevich [5] for testing k-colorability. We next give a high-level description of the analysis.

Given the sample S, we construct a k-ary tree. Each node in the tree corresponds to a partial partition of the sample. That is, a partition of a subset of the sample. In particular, each internal node corresponds to a *valid* partition (where the root corresponds to a trivial partition of the empty set). If an internal node corresponds to a partition $(S'_1, \ldots, S'_k)$ of a subset S' of the sample, then its children correspond to all partitions of $S' \cup \{u\}$ that extend the partition $(S'_1, \ldots, S'_k)$ for some sample point $u \in S \setminus S'$. That is, partitions of the form $(S'_1, \ldots, S'_{i-1}, S'_i \cup \{u\}, S'_{i+1}, \ldots, S'_k)$. Observe that if we obtain a tree for which all leaves correspond to invalid partitions (i.e., that violates some constraint of P), then there is no valid partition of S.

Consider a node x in the tree, corresponding to a partition $(S'_1, \ldots, S'_k)$ of $S' \subset S$. For each vertex $v \notin S'$, let $0 \leq a_x(v) \leq k$ be the number of parts in the partition to which u can be added so that the resulting partition is valid, and let a_x be the sum of $a_x(v)$ taken over all $v \notin S'$. Observe that for the root of the tree, r (which corresponds to $S' = \emptyset$), $a_r = k \cdot n$, and

[5] If the tested graph partition property is NP-hard to decide, then the running time is super-polynomial in the sample size, which is unavoidable assuming $P \neq NP$.

if y is a child of x, then $a_y \leq a_x$. If the partition corresponding to y is invalid, then $a_y = 0$. We show that with high constant probability over the choice of the sample, we can construct a tree for which the following holds. For every path in the tree, the value of a_x decreases in a relatively significant manner when comparing each node to its children. This allows us to show that we can obtain a tree in which all partitions corresponding to the leaves are invalid.

1.3 Easily Testable Graph Partition Properties Must be in $\mathcal{GPP}_{0,1}$

Our proof that if a property $P \in \mathcal{GPP}$ is easily testable, then P must be in $\mathcal{GPP}_{0,1}$ is the most technically involved part of this work. The proof consists of several steps, and we next give a high-level outline of these steps. We note that the proof uses the fact that easy testability implies strong testability. That is, we rely on the existence of a one-sided error tester for the property that is oblivious of the size of the graph, but we do not rely on the tester having complexity $\text{poly}\left(\frac{1}{\epsilon}\right)$.

Recall that properties in $\mathcal{GPP}_{0,1}$ are defined by the following types of constraints over graph partitions. First, for each part, either the edge density within the part is unconstrained, or it is constrained in an extreme manner. The latter means that no edges are allowed within the part, or that there must be all possible edges. We say in such a case that the part is *homogeneous*. Similarly, for each pair of parts, either there is no constraint on the edge density between the parts, or it is extreme (no edges, or all edges). Here too we say in the latter case that the pair is homogeneous. Finally, as opposed to $\mathcal{GPP}$, there are no constraints on the sizes of the parts. Observe that the trivial property, that is, the property that contains all graphs, belongs to $\mathcal{GPP}_{0,1}$ (since it can be defined by a single part with no edge-density constraints).

In what follows, for a property $P \in \mathcal{GPP}$ and a graph $G = (V, E)$ satisfying P, a partition $(V_1, \ldots, V_k)$ of V is said to be a *witness partition* with respect to P, if in G, $(V_1, \ldots, V_k)$ satisfies the constraints imposed by P. We say in such a case that the pair $(G, (V_1, \ldots, V_k))$ satisfies P. We first prove that if P is easily testable, then either it is trivial, or for every graph G satisfying P and witness partition $(V_1, \ldots, V_k)$, all parts are homogeneous. This is established by showing that if there exists a graph G in P with a witness partition that has some non-homogeneous part, then the premise that P is easily testable implies that P is trivial. The proof uses a type of "multiplying" operation on the graph G. Once we have only homogeneous parts, we can also establish the homogeneity of pairs (among those that are constrained in terms of edge-density). At this point it remains to show that there can be no size constraints on the parts.

To this end we prove a dichotomy claim. Let P' be the same property as P except that there are no size constraints. The claim is that either $P = P'$ or P' is in a certain sense far from P. We then show that the second case cannot hold if P is easily testable. In order to prove the dichotomy claim, we define a certain mathematical program, that, roughly speaking, is related to modifications of graph-partition pairs that satisfy P' to graph-partition pairs that satisfy P (by "fixing" the size constraints). In particular, the existence of a feasible solution corresponds to $P = P'$. On the other hand, if there is no feasible solution, then we show that P' is far from P. This proof involves a probabilistic construction of a graph that satisfies P but is sufficiently far from satisfying P'.

1.4 Related Work

Easily testable graph properties

Besides the class of graph partition properties, there are several results characterizing the set of easily testable graph properties among other classes. Alon [1] proved that the property of being H-free is easily testable if and only if H is bipartite. Alon and Shapira [7] proved that for any graph H besides P_2, P_3, P_4, C_4 and their complements, the property of being induced H-free is not easily testable. It was also shown in [7, 4] that induced H-freeness is easily-testable for P_2, P_3, P_4 and their complements, and the case of C_4 (and its complement) is the only one that remains open. In addition, the graph properties perfectness and comparability were shown to be not easily testable [4]. Gishboliner and Shapira [11] recently made significant progress by providing sufficient and necessary conditions for guaranteeing that a hereditary graph property is easily testable, implying all the positive and negative results mentioned above. It is worth noting, however, that their criteria do not apply to many properties in $\mathcal{GPP}_{0,1}$ (for example, (p,q)-colorability), that are shown to be easily testable in our work.

Testing properties in $\mathcal{GPP}_{NR}$ with one sided error

As mentioned previously, Goldreich and Trevisan [13] studied the one-sided error testability of $\mathcal{GPP}_{NR}$. They showed that every strongly testable property in $\mathcal{GPP}_{NR}$ belongs to a class of properties that generalizes k-colorability. Each property P in this class is defined by a set of pairs $A_P = \{(i,j)\,|0 \leq i,j \leq k\}$, where the property P is the set of k-colorable graphs with the additional constraint that if $(i,j) \in A_P$, then there are no edges between the vertices with color i and the vertices with color j. In addition, the property of being a clique and the property containing all graphs are both stongly testable graph properties in $\mathcal{GPP}_{NR}$.

We build on [13]'s technique of multiplying a graph-partition pair to derive the fact that all the easily testable properties in $\mathcal{GPP}$ only have homogeneous constraints on the edge density within and between parts. The idea of finding assignments to variables corresponding to moving vertices between parts also appears in [13], but they did not have to optimize over a mathematical program, and the assignments they defined could be used straightforwardly to establish the equivalence between a property and its relaxation. One of the main ideas used in [13] to derive the characterization was showing that strongly testable properties in $\mathcal{GPP}_{NR}$ are closed under removal of edges (except for the property of being a clique), and they rely on this fact heavily when deriving the implication regarding the assignments they define and when performing the multiplication. We could not use this idea as it does not hold in our case, because the existence of relative edge bounds in $\mathcal{GPP}$ enables easily testable properties to have lower bounds on the number of edges between or within parts. This is basically the main reason why the set of easily testable properties in $\mathcal{GPP}$ has a richer structure than in $\mathcal{GPP}_{NR}$. This is why we had to use notions that do not appear in their analysis such as weak and strong violations of assignments and rely on the probabilistic method to establish our result.

Organization

In Section 2 we expand on the result regarding the "only if" direction of Theorem 3, that all the easily testable properties in $\mathcal{GPP}$ are in $\mathcal{GPP}_{0,1}$. All the details regarding the other two results can be found in the accompanying full version of the paper, as well as the proofs of all claims stated in Section 2.

2 Easily Testable Graph Partition Properties Must be in $\mathcal{GPP}_{0,1}$

As stated in Theorem 3, a property $P \in \mathcal{GPP}$ is easily testable if and only if $P \in \mathcal{GPP}_{0,1}$. In this section we provide the proof structure and some of the proof details for the claim that if a graph partition property is easily testable, then $P \in \mathcal{GPP}_{0,1}$. We first define the concept of a t-multiplier, which is used several times throughout the proof. In what follows, given a graph $G = (V, E)$ and a set of vertices $U \subseteq V$ we denote by $G[U]$ the subgraph induced by U.

2.1 t-Multipliers

▶ **Definition 4.** Let $G = (V, E)$ be a graph over n vertices and let $(V_1, ..., V_k)$ be a partition of V. For an integer t we say that a graph-partition pair $(G', (V'_1, ..., V'_k))$ is a ***t*-multiplier** of $(G, (V_1, ..., V_k))$, if the following holds (where $G' = (V', E')$).
Vertices: $|V'| = t \cdot n$.
Partition: For each $1 \leq i \leq k$, $|V'_i| = t \cdot |V_i|$.
Within Edges: Suppose $G[V_i]$ has $\alpha_{ii} |V_i|^2$ edges. Then $G'[V'_i]$ has $\alpha_{ii} t^2 |V_i|^2$ edges.
Between Edges: Suppose G has $2\alpha_{ij} |V_i| \cdot |V_j|$ edges between V_i and V_j. Then G' has $2\alpha_{ij} \cdot t^2 |V_i| \cdot |V_j|$ edges between V'_i and V'_j.

Recall that given a graph $G = (V, E)$ and a partition $(V_1, ..., V_k)$ of V that satisfies the constraints imposed by property P in $\mathcal{GPP}$, we say that $(V_1, ..., V_k)$ is a *witness* partition to the fact that G satisfies P. In short, we say that the graph-partition pair $(G, (V_1, ..., V_k))$ *satisfies* P. The next claim trivially holds.

▶ **Claim 5.** *If $(G, (V_1, ..., V_k))$ is a graph-partition pair satisfying P and $(G', (V'_1, ..., V'_k))$ is a t-multiplier of $(G, (V_1, ..., V_k))$, then the pair $(G', (V'_1, ..., V'_k))$ also satisfies P.*

2.2 Easily Testable Graph Partition Properties are Homogeneous

Let P be a graph partition property. We note that even if there are no explicit bounds on a part's size or on the edge density within a part or between a pair of parts, such constraints may be implicitly induced by the combination of other constraints. This leads to the following definition.

▶ **Definition 6.** Given a pair of integers $(i, j) \in [k] \times [k]$, we say that P has **no constraints on the edge density between the parts** (i, j) if for every graph-partition pair $(G, (V_1, ..., V_k))$ satisfying P, any graph G', obtained from G by performing arbitrary vertex-pair modifications between V_i and V_j in G, satisfies P and $(V_1, ..., V_k)$ serves as a witness partition. Otherwise, we say that P **constrains the edge density between parts** (i, j). For the special case where $i = j$ we say that P **constrains the edge density within part** i.

We say that a graph is *homogeneous* if it is either an independent set or a clique. We can classify the properties in $\mathcal{GPP}$ into two sets, corresponding to the following complementary two cases.
Case (a): There exists a graph-partition pair $(G, (V_1, ..., V_k))$ satisfying P and an integer $1 \leq i \leq k$ such that $G[V_i]$ is non-homogeneous.
Case (b): For every graph-partition pair $(G, (V_1, ..., V_k))$ that satisfies P it holds that $G[V_i]$ is homogeneous for every $1 \leq i \leq k$.
We note that Case (b) can be shown to imply a stronger statement. If Case (b) holds, then not only every part is homogeneous, but rather the following holds: For each $1 \leq i \leq k$,

either for every graph-partition pair $(G,(V_1,...,V_k))$ satisfying the property, $G[V_i]$ is an independent set, or for every graph-partition pair $(G,(V_1,...,V_k))$ satisfying the property, $G[V_i]$ is a clique. We next establish the following implication of Case (a).

▶ **Claim 7.** *Let P be an easily testable graph partition property for which Case (a) holds. Then for every proximity parameter ϵ, every graph is ϵ-close to satisfying P.*

Proof. Since Case (a) holds, there exists a graph-partition pair $(G,(V_1,...,V_k))$ satisfying P and an integer $1 \leq i \leq k$ such that $G[V_i]$ is non-homogeneous. Suppose by way of contradiction that there exists a one-sided error tester for P that is input-size oblivious. Denote the tester by $\mathcal{T}$. By [2, 13], we can assume without loss of generality that the algorithm $\mathcal{T}$ makes its decision based on an inspection of the subgraph induced by a random sample of s_ϵ vertices chosen independently and uniformly at random, where s_ϵ is a function of ϵ and is independent of n.

By Claim 5, for every $(G',(V'_1,...,V'_k))$ that is an s_ϵ-multiplier of $(G,(V_1,...,V_k))$, we have that G' satisfies P (with the witness partition $(V'_1,...,V'_k)$). Therefore, $\mathcal{T}$ must accept each such G' with probability 1. We next show that for every graph H over at most s_ϵ vertices, there exists at least one such graph G' for which $G'[V'_i]$ contains H as an induced subgraph. The claim will then follow since the tester must accept given any induced subgraph that it observes, implying that it accepts all graphs with probability 1.

Let $|V_i| = n_i$ and let $v_i^1,\ldots,v_i^{s_\epsilon n_i}$ denote the vertices in V'_i. Observe that since $G[V_i]$ has at least one edge and at least one non-edge, for every $(G',(V'_1,...,V'_k))$ that is an s_ϵ-multiplier of $(G,(V_1,...,V_k))$, it holds that $G'[V'_i]$ has $m'_i \geq s_\epsilon^2$ edges and $(t\cdot n_i)^2 - m'_i \geq s_\epsilon^2$ non-edges. Let H be some fixed graph over $s \leq s_\epsilon$ vertices with m_H edges (and $s^2 - m_H$ non-edges). Since $m_H \leq s_\epsilon^2 \leq m'_i$ and $s^2 - m_H \leq s_\epsilon^2 \leq (t\cdot n_i)^2 - m'_i$, the definition of an s_ϵ-multiplier allows to let the subgraph of $G'[V'_i]$ induced by the vertices $v_i^1,\ldots,v_i^s$ be H.

That is, the tester $\mathcal{T}$ accepts every graph with probability 1, and hence, for every ϵ, every graph is ϵ-close to satisfying P. ◀

We emphasize that Claim 7 holds for every graph and not only for sufficiently large graphs. It follows that if P is an easily testable graph partition property that satisfies Case (a) then P is in fact the trivial graph partition property that contains all graphs. This property clearly belongs to $\mathcal{GPP}_{0,1}$. Hence, from now on we can assume that Case (b) holds. In other words, we consider properties $P \in \mathcal{GPP}$ for which every graph satisfying P can only be validly partitioned in such a way that every part of the partition is homogeneous. That is, if G is a graph satisfying P and $(V_1,...,V_k)$ is a witness partition, then for every pair $(i,j) \in [k]\times[k]$, the subgraph $G[V_i \cup V_j]$ is either a split graph, or a bipartite graph or a cobipartite graph. We can use this fact together with an application of an appropriate multiplier to establish the following claim.

▶ **Claim 8.** *Let P be an easily testable property in $\mathcal{GPP}$. If there exists a graph-partition pair $(G,(V_1,...,V_k))$ that satisfies P such that the edge density between a pair of parts is neither 0 nor 1, then P has no constraints on the edge density between the two parts.*

In this subsection we showed that if a graph partition property P is easily testable and P constrains the edge density within a particular part, then the part must be homogeneous. (To be precise, P either forces the part to be an independent set or it forces it to be a clique.) Similarly, if P constrains the edge density between a pair of parts then it either forces the edge density within the pair to be 0 or it forces it to be 1. We call such properties *homogeneous graph partition properties*. That is, a homogeneous graph partition property is defined similarly to a property in $\mathcal{GPP}_{0,1}$ except that unlike $\mathcal{GPP}_{0,1}$, a homogeneous graph

partition property possibly has size constraints on the parts. In the next subsection we show that if such a property is easily testable, then it has no size constraints.

2.3 No Constraints on the Sizes of Parts

Suppose P is a homogeneous property in $\mathcal{GPP}$ that possibly has size constraints in its specification and is easily testable. We show that if this is the case, then there exists an equivalent formulation of P that has no size constraints. Namely, we show that the simple relaxation of P whose specification contains the same homogeneity constraints as those specified by P but excludes its size constraints, is equivalent to P (under the assumption that P is easily testable). From now on, given a graph partition property P, we denote the relaxation of P (obtained by deleting the size constraints) by P'.

In order to obtain the above we prove a *dichotomy of properties*. In particular, we show that every homogeneous graph partition property P falls into one of two disjoint categories.

▶ **Claim 9** (The Dichotomy of Properties). *Every homogeneous property P in the class $\mathcal{GPP}$ satisfies one of the following:*

1. $P = P'$.
2. *There exists $\epsilon > 0$ such that for every n_0 there exists a graph $G' \in P'$ of size $n > n_0$ where G' is ϵ-far from P.*

The implication of the *dichotomy of properties* is that a homogeneous graph partition property P with size constraints cannot be close to its relaxation P'. Either P is equivalent to P' or P is far from P' (for an appropriate distance measure). We claim that if the latter case holds then P is not easily-testable. Namely,

▶ **Claim 10.** *Suppose there exists $\epsilon > 0$ such that for every n_0 there exists a graph $G' \in P'$ of size $n > n_0$ where G' is ϵ-far from P. Then P is not easily testable.*

Combining Claim 10 with *the dichotomy of properties* (Claim 9) establishes the following: If a graph partition property P enforces size constraints, then P is not easily testable. It still remains to prove that *the dichotomy of properties* indeed holds. In order to do so, we first define the notions of a size vector and of a property's set of assignments. Then we show the existence of another dichotomy, *the trivial dichotomy*, which, as we state below, implies *the dichotomy of properties*.

▶ **Definition 11.** Given a homogeneous property P in $\mathcal{GPP}$ we define a set of variables $X = \{x_{ij} | (i,j) \in [k] \times [k]\}$. For a size vector $\vec{\rho}$, we say that an assignment $\varphi : X \to [0,1]$ is **sizewise valid** if: $\forall i' \in [k] : \sum_{i=1}^{k} \varphi(x_{i'i}) = \rho_{i'}$ and $\forall i \in [k] : \rho_i^L \leq \sum_{i'=1}^{k} \varphi(x_{i'i}) \leq \rho_i^U$.

We interpret an assignment φ as a transformation from a size vector $\vec{\rho}$ that violates the size constraints of P to a size vector that satisfies the size constraints. In particular, we interpret $\varphi(x_{ij})$ as the fraction of vertices (relative to n) that should be transferred from part i to part j (or stay in part i if $i = j$) in order to satisfy the size constraints imposed by P. Applying the transformation induced by a sizewise valid assignment φ to a graph G' that satisfies P' clearly results in a graph G that satisfies the size constraints imposed by P. However, the assignment being sizewise valid does not necessarily induce a transformation resulting in a graph that satisfies the edge density constraints. This leads to the notion of a *violation* defined next.

▶ **Definition 12.** Given a homogeneous property P in $\mathcal{GPP}$ and an assignment φ we say that a pair of variables $\{x_{i'i}, x_{j'j}\}$ constitutes a **violation** in the assignment φ with respect to P if $\varphi(x_{i'i}) \neq 0, \varphi(x_{j'j}) \neq 0, d_P(i,j) \neq \perp, d_P(i',j') \neq d_P(i,j)$. If $d_P(i',j') = \perp$, then we say that the violation is **weak**. Otherwise, we say the violation is **strong**.

The situation stated in the above definition is considered to be a violation of the edge density constraints because it can be interpreted as vertices being transferred to a pair of parts whose edge density differs from that of the pair of parts those vertices originated from. Given a violation $\{x_{i'i}, x_{j'j}\}$ we define the violation's *size* as $\min\{\varphi(x_{i'i}), \varphi(x_{j'j})\}$. We call a violation of size at least δ a *δ-violation*.

The following claim describes *the trivial dichotomy*.

▶ **Proposition 13** (The Trivial Dichotomy). *Let P be a homogeneous graph partition property. Exactly one of the following holds:*

1'. *For every size vector $\vec{\rho}$ there exists a sizewise valid assignment φ which is free of violations with respect to P.*

2'. *There exists a size vector $\vec{\rho}$ for which every sizewise valid assignment φ contains a violation with respect to P.*

2.3.1 The Trivial Dichotomy Implies The Dichotomy of Properties

The trivial dichotomy (Proposition 13) trivially holds as the second case is the complement of the first. In order to prove that *the trivial dichotomy* implies *the dichotomy of properties* (Claim 9) we have to prove that Case 1' in *the trivial dichotomy* implies Case 1 in *the dichotomy of properties* and that Case 2' in *the trivial dichotomy* implies Case 2 in *the dichotomy of properties*. We next give the high-level idea for the proof that Case 2' implies Case 2.

Let $\vec{\rho}$ be a size vector for which every assignment φ that is sizewise valid contains a violation. We define a set of decision variables: $Y = \{y_{i'i} | 1 \leq i \leq k, 1 \leq i' \leq k\}$.

▶ **Definition 14.** We say that a pair of decision variables $y_{i'i}, y_{j'j} \in Y$ are in potential conflict if $d_P(i,j) \neq \perp$ and $d_P(i',j') \neq d_P(i,j)$.

That is, $y_{i'i}$ is in potential conflict with $y_{j'j}$ if the pair $\{x_{i'i}, x_{j'j}\}$ constitutes a violation with respect to P in an assignment φ where $\varphi(x_{i'i}) > 0$ and $\varphi(x_{j'j}) > 0$.

We define a mathematical program on the set of decision variables Y. The objective of the program is to minimize the function $\max_{(y_{i'i}, y_{j'j}) \in Y} \{\min\{y_{i'i}, y_{j'j}\}\}$.

The feasible region is defined by the following system of linear constraints:

$$\forall i' \in [k] : \sum_{i=1}^{k} y_{i'i} = \rho_{i'}$$

$$\forall i \in [k] : \rho_i^L \leq \sum_{i'=1}^{k} y_{i'i} \leq \rho_i^U$$

$$\forall (i',i) \in [k] : 0 \leq y_{i'i} \leq 1$$

There is a one-to-one correspondence between feasible solutions to the program and sizewise valid assignments. Moreover, the correspondence between assignments and feasible solutions has the property that an assignment is free of violations if and only if the objective function attains a value of 0 for the corresponding feasible solution. It is shown in the full version that there exists a feasible solution (and a corresponding assignment φ) that minimizes $\max_{(y_{i'i}, y_{j'j}) \in Y} \{\min\{y_{i'i}, y_{j'j}\}\}$. Denote by δ the value of the objective function under that solution. Since we have assumed that every sizewise valid assignment φ contains a violation, there is no feasible solution attaining an objective function value of 0. In other words, $\delta > 0$. Therefore, every sizewise valid assignment contains a violation of size at least δ.

Let $\epsilon = \frac{1}{16}\delta^2$. We show that for every n_0 there exists a graph of size $n > n_0$ that is ϵ-far from P. Here we show this for the case where there are only weak δ-violations. We construct a random graph $G' = (V', E')$ of size n and prove that with positive probability G' is ϵ-far from every n-size graph G satisfying P. Let $V' = \{1, ..., n\}$ be the set of vertices of the graph G'. We arbitrarily partition $V[G']$ into k disjoint sets $U_1, ..., U_k$ in such a way that the number of vertices in U_i is $\rho_i n$. Let E' be defined as follows: For each pair of vertices $u \in U_i, v \in U_j$: If $d_P(i,j) = 0$ we do not connect u and v. If $d_P(i,j) = 1$ we do connect u and v. If $d_P(i,j) = \perp$ we connect u and v with probability $\frac{1}{2}$.

For each $1 \leq i \leq k$ where $d_P(i,i) = \perp$ we define the event $\mathcal{E}_i$ as follows: *For every set* $A \subseteq U_i$ *of size* $|A| \geq \delta n$ *it holds that* $e_{G'}(A,A) \geq \epsilon n^2$ *and* $\overline{e}_{G'}(A,A) \geq \epsilon n^2$.

For each pair $(i,j) \in [k] \times [k]$ where $d_P(i,j) = \perp$ we define the event $\mathcal{E}_{ij}$ as follows: *For every pair of disjoint sets* $(A,B) \subseteq U_i \times U_j$ *s.t.* $A \subseteq U_i, B \subseteq U_j$ *where* $|A| \geq \delta n$ *and* $|B| \geq \delta n$ *it holds that* $e_{G'}(A,B) \geq \epsilon n^2$ *and* $\overline{e}_{G'}(A,B) \geq \epsilon n^2$..

Note that the event $\mathcal{E}_i$ is not the same as event $\mathcal{E}_{ii}$. We denote by $\mathcal{E}$ the conjunction of all the events defined above. A detailed probabilistic analysis establishes the following claim.

▶ **Claim 15.** $\Pr[\mathcal{E}] > 0$.

Claim 15 implies the existence of a graph G^* for which $\mathcal{E}$ holds and the pair $(G^*, (V_1, ..., V_k))$ satisfies P'. We next show that G^* is ϵ-far from every n-vertex graph G satisfying P. Let $(G, (V_1, ..., V_k))$ be any graph-partition pair satisfying P where G is a graph over n vertices. We define the assignment φ as $\varphi(x_{i'i}) = \frac{|U_{i'} \cap V_i|}{n}$. Since the assignment φ can be shown to be sizewise valid, it contains a δ-violation $\{x_{i'i}, x_{j'j}\}$. Suppose the violation is weak. That is, $d_P(i',j') = \perp$ and $d_P(i,j) \neq \perp$. The fact that $\varphi(x_{i'i}) \geq \delta$ and $\varphi(x_{j'j}) \geq \delta$ implies that $|U_{i'} \cap V_i| \geq \delta n$ and $|U_{j'} \cap V_j| \geq \delta n$. The event $\mathcal{E}_{i'j'}$ holds and so do $\mathcal{E}_{i'}$ and $\mathcal{E}_{j'}$. Hence, $e_{G^*}(V_i, V_j) \geq \epsilon n^2$ and $\overline{e}_{G^*}(V_i, V_j) \geq \epsilon n^2$. However, since $d_P(i,j) \neq \perp$, the number of vertex-pair modifications required in order to obtain G from G^* is at least ϵn^2. In other words, the graph G^* is ϵ-far from every graph G that satisfies P.

It follows that easily-testable homogeneous graph partition properties cannot have size constraints, and if they do, then these size constraints are in fact redundant. In conclusion, if a graph partition property P is easily-testable then it is both homogeneous and it has no size constraints. That is, $P \in \mathcal{GPP}_{0,1}$.

References

1. Noga Alon. Testing subgraphs in large graphs. *Random Structures & Algorithms*, 21(3-4):359–370, 2002.
2. Noga Alon, Eldar Fischer, Michael Krivelevich, and Mario Szegedy. Efficient testing of large graphs. *Combinatorica*, 20(4):451–476, 2000.
3. Noga Alon, Eldar Fischer, Ilan Newman, and Asaf Shapira. A combinatorial characterization of the testable graph properties: it's all about regularity. *SIAM Journal on Computing*, 39(1):143–167, 2009.
4. Noga Alon and Jacob Fox. Easily testable graph properties. *Combinatorics, Probability and Computing*, 24(4):646–657, 2015.
5. Noga Alon and Michael Krivelevich. Testing k-colorability. *SIAM Journal on Discrete Mathematics*, 15(2):211–227, 2002.
6. Noga Alon and Asaf Shapira. Testing subgraphs in directed graphs. *Journal of Computer and System Sciences*, 69(3):354–382, 2004.
7. Noga Alon and Asaf Shapira. A characterization of easily testable induced subgraphs. *Combinatorics, Probability and Computing*, 15(6):791–805, 2006.

8 Noga Alon and Asaf Shapira. A characterization of the (natural) graph properties testable with one-sided error. *SIAM Journal on Computing*, 37(6):1703–1727, 2008.

9 Christian Borgs, Jennifer Chayes, László Lovász, Vera T Sós, Balázs Szegedy, and Katalin Vesztergombi. Graph limits and parameter testing. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 261–270. ACM, 2006.

10 David Conlon and Jacob Fox. Graph removal lemmas. *Surveys in combinatorics*, 1(2):3, 2013.

11 Lior Gishboliner and Asaf Shapira. Removal lemmas with polynomial bounds. *arXiv preprint arXiv:1611.10315*, 2016.

12 Oded Goldreich, Shari Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM (JACM)*, 45(4):653–750, 1998.

13 Oded Goldreich and Luca Trevisan. Three theorems regarding testing graph properties. *Random Structures & Algorithms*, 23(1):23–57, 2003.

On Closeness to k-Wise Uniformity

Ryan O'Donnell
Carnegie Mellon University, Pittsburgh, PA, USA
odonnell@cs.cmu.edu

Yu Zhao
Carnegie Mellon University, Pittsburgh, PA, USA
yuzhao1@cs.cmu.edu

Abstract

A probability distribution over $\{-1,1\}^n$ is (ϵ,k)-*wise uniform* if, roughly, it is ϵ-close to the uniform distribution when restricted to any k coordinates. We consider the problem of how far an (ϵ,k)-wise uniform distribution can be from any globally k-wise uniform distribution. We show that every (ϵ,k)-wise uniform distribution is $O(n^{k/2}\epsilon)$-close to a k-wise uniform distribution in total variation distance. In addition, we show that this bound is optimal for all even k: we find an (ϵ,k)-wise uniform distribution that is $\Omega(n^{k/2}\epsilon)$-far from any k-wise uniform distribution in total variation distance. For $k=1$, we get a better upper bound of $O(\epsilon)$, which is also optimal.

One application of our closeness result is to the sample complexity of testing whether a distribution is k-wise uniform or δ-far from k-wise uniform. We give an upper bound of $O(n^k/\delta^2)$ (or $O(\log n/\delta^2)$ when $k=1$) on the required samples. We show an improved upper bound of $\tilde{O}(n^{k/2}/\delta^2)$ for the special case of testing fully uniform vs. δ-far from k-wise uniform. Finally, we complement this with a matching lower bound of $\Omega(n/\delta^2)$ when $k=2$.

Our results improve upon the best known bounds from [3], and have simpler proofs.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms

Keywords and phrases k-wise independence, property testing, Fourier analysis, Boolean function

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.54

Related Version A full version of this paper is available at `https://arxiv.org/abs/1806.03569`.

Funding 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).

1 Introduction

1.1 k-wise uniformity and almost k-wise uniformity

We say that a probability distribution over $\{-1,1\}^n$ is k-*wise uniform* if its marginal distribution on every subset of k coordinates is the uniform distribution. For Fourier analysis of the Hamming cube, it is convenient to identify the distribution with its density function $\varphi : \{-1,1\}^n \to \mathbb{R}^{\geq 0}$ satisfying

$$\mathop{\mathbf{E}}_{\boldsymbol{x}\sim\{-1,1\}^n}[\varphi(\boldsymbol{x})] = 1.$$

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018).
Editors: Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer; Article No. 54; pp. 54:1–54:19

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

We write $\boldsymbol{x} \sim \varphi$ to denote that $\boldsymbol{x}$ is a random variable drawn from the associated distribution with density φ:

$$\mathop{\mathbf{Pr}}_{\boldsymbol{x}\sim\varphi}[\boldsymbol{x} = x] = \frac{\varphi(x)}{2^n}$$

for any $x \in \{-1,1\}^n$. Then a well-known fact is that a distribution is k-wise uniform if and only if the Fourier coefficient of φ is 0 on every subset $S \subseteq [n]$ of size between 1 and k:

$$\widehat{\varphi}(S) = \mathop{\mathbf{E}}_{\boldsymbol{x}\sim\varphi}\left[\prod_{i\in S} \boldsymbol{x}_i\right] = 0.$$

k-wise uniformity is an essential tool in theoretical computer science. Its study dates back to work of Rao [25]. They studied k-wise uniform sets, which are special cases of k-wise uniform distribution. A subset of $\{-1,1\}^n$ is *a k-wise uniform set* if the uniform distribution on this subset is k-wise uniform. Rao gave constructions of a pairwise-uniform set of size $n+1$ (when $n = 2^r - 1$ for any integer r), a 3-wise uniform set of size $2n$ (when $n = 2^r$ for any integer r), and a lower bound (reproved in [4, 14]) that a k-wise uniform set on $\{-1,1\}^n$ requires size at least $\Omega(n^{\lfloor k/2 \rfloor})$. An alternative proof of the lower bound for even k is shown in [6] using a hypercontractivity-type technique, as opposed to the linear algebra method. Coding theorists have also heavily studied k-wise uniformity, since MacWilliams and Sloane showed that linear codes with dual minimum distance $k+1$ correspond to k-wise uniform sets in [21]. The importance in theoretical computer science of k-wise independence for derandomization arose simultaneously in many papers, with [17, 20] emphasizing derandomization via the most common pairwise-uniformity case, and [4, 14] emphasizing derandomization based on k-wise independence more generally.

A distribution is "almost k-wise uniform" if its marginal distribution on every k coordinates is very close to the uniform distribution. Typically we say two distributions φ, ψ are *δ-close*, if the total variation distance between φ and ψ is at most δ; and we say they are *δ-far*, if the total variation distance between them is more than δ. However the precise notion of "close to uniform" has varied in previous work. Suppose ψ is the density function for the marginal distribution of φ restricted to some specific k coordinates and $\mathbf{1}$ is the density function for the uniform distribution. Several standard ways are introduced in [6, 3] to quantify closeness to uniformity, corresponding to the L_1, L_2, L_∞ norms:

- (L_1 norm): $\|\psi - \mathbf{1}\|_1 = 2d_{\mathrm{TV}}(\psi, \mathbf{1}) \le \epsilon$, where d_{TV} denotes total variation distance;
- (L_2 norm): $\|\psi - \mathbf{1}\|_2 = \sqrt{\chi^2(\psi, \mathbf{1})} = \sqrt{\sum_{S\neq\emptyset} \widehat{\psi}(S)^2} \le \epsilon$, where $\chi^2(\psi, \mathbf{1})$ denotes the χ^2-divergence of ψ from the uniform distribution;
- (L_∞ norm): $\|\psi - \mathbf{1}\|_\infty \le \epsilon$, or in other words, for any $x \in \{-1,1\}^n$,

$$\left|\mathop{\mathbf{Pr}}_{\boldsymbol{x}\sim\psi}[\boldsymbol{x} = x] - 2^{-k}\right| \le 2^{-k}\epsilon.$$

Note the following: First, closeness in L_1 norm is the most natural for algorithmic derandomization purposes: it tells us that the algorithm cannot tell ψ is different from the uniform distribution up to ϵ error. Second, these definitions of closeness are in increasing order of strength. On the other hand, we also have that $\|\psi - \mathbf{1}\|_1 \le \|\psi - \mathbf{1}\|_\infty \le 2^k \|\psi - \mathbf{1}\|_1$; thus all these notions are within a factor of 2^k. We generally consider k to be constant (or at worst, $O(\log n)$), so that these notions are roughly the same.

A fourth reasonable notion, proposed by Naor and Naor in [22], is that the distribution has a small bias over every non-empty subset of at most k coordinates. We say density

function φ is (ϵ, k)-*wise uniform* if for non-empty set $S \subseteq [n]$ with size at most k,

$$|\widehat{\varphi}(S)| = \left| \mathop{\mathbf{Pr}}_{\boldsymbol{x}\sim\varphi}\left[\prod_{i\in S} \boldsymbol{x}_i = 1\right] - \mathop{\mathbf{Pr}}_{\boldsymbol{x}\sim\varphi}\left[\prod_{i\in S} \boldsymbol{x}_i = -1\right]\right| \leq \epsilon.$$

Here we also have $\epsilon = 0$ if and only if φ is exactly k-wise uniform. Clearly if the marginal density of φ over every k coordinates is ϵ-close to the uniform distribution in total variation distance, then φ is (ϵ, k)-wise uniform. On the other hand, if φ is (ϵ, k)-wise uniform, then the marginal density of φ over every k coordinates is $2^{k/2}\epsilon$-close to uniform distribution in total variation distance. Again, if k is considered constant, this bias notion is also roughly the same as previous notions. In the rest of paper we prefer this (ϵ, k)-wise uniform notion for "almost k-wise uniform" because of its convenience for Fourier analysis.

The original paper about almost k-wise uniformity is [22], which is concerned with derandomization; e.g., they use (ϵ, k)-wise uniformity for derandomizing the "set balancing (discrepancy)" problem. Alon et al. give a further discussion of the relationship between almost k-wise uniformity and derandomization in [6]. The key idea is the following: In many cases of randomized algorithms, the analysis only relies on the property that the random bits are k-wise uniform, as opposed to fully uniform. Since there exists an efficiently samplable k-wise uniform distribution on a set of size at most $O(n^{\lfloor k/2 \rfloor})$, one can reduce the number of random unbiased bits used in the algorithm down to $O(k \log n)$. To further reduce the number of random bits used, a natural line of thinking is to consider distributions which are "almost k-wise uniformity". Alon et al. [5] showed that we can deterministically construct (ϵ, k)-wise uniform sets that are of size $\mathrm{poly}(2^k, \log n, 1/\epsilon)$, much smaller than exact k-wise uniform ones (roughly $\Omega(n^{\lfloor k/2 \rfloor})$ size). Therefore we can use substantially fewer random bits by taking random strings from an almost k-wise uniform distribution.

However we need to ensure that the original analysis of the randomized algorithm still holds under the almost k-wise uniform distribution. This is to say that if the randomized algorithm behaves well on a k-wise uniform distribution, it may also work as well with an (ϵ, k)-wise uniform distribution, when the parameter ϵ is small enough.

1.2 The Closeness Problem

For the analysis of derandomization, it would be very convenient if (ϵ, k)-wise uniformity – which means that "every k-local view looks close to uniform" – implies global δ-closeness to k-wise uniformity. A natural question that arises, posed in [6], is the following:

How small can δ be, such that the following is true: For every (ϵ, k)-wise uniform distribution φ on $\{-1, 1\}^n$, φ is δ-close to some k-wise uniform distribution?

In this paper, we will refer to this question as *the Closeness Problem*.

1.2.1 Previous work and applications

On one hand, the main message of [6] showed a lower bound: For every even constant $k > 4$, they gave an (ϵ, k)-wise uniform distribution with $\epsilon = O(1/n^{k/4-1})$, yet which is $\frac{1}{2}$-far from every k-wise uniform distribution in total variation distance.

On the other hand, [6] proved a very simple theorem that $\delta \leq O(n^k\epsilon)$ always holds. Despite simplicity, this upper bound has been used many times in well known results.

One application is in circuit complexity. [6]'s upper bound is used for fooling disjunctive normal formulas (DNF) [11] and AC^0 [12]. In these works, once the authors showed that k-wise uniformity suffices to fool DNF/AC^0, they deduced that $(O(1/n^k), k)$-uniform distributions

suffice, and hence $O(1/n^k)$-biased sets sufficed trivially. [6]'s upper bound is also used as a tool for the construction of two-source extractors for a similar reason in [13, 19].

The notions of pairwise-uniformity, k-wise uniformity, and δ-closeness to k-wise uniformity are also important for hardness of constraint satisfactory problems (CSPs). Austrin and Mossel [7] shows that one can obtain integrality gaps and UGC-hardness for CSPs based on k-wise uniform distributions of small support size. If a predicate is k-wise uniform, Kothari et al. [18] showed that one can get SOS-hardness of refuting random instances of it when there are around $n^{(k+1)/2}$ constraints. Indeed, [18] shows that if we have a predicate that is δ-close to k-wise uniform, then with roughly $n^{(k+1)/2}$ random constraints, SOS cannot refute that a $(1 - O(\delta))$-fraction of constraints are satisfiable. This also motivates studying δ-closeness to k-wise uniformity, and how it relates to Fourier coefficients. δ-closeness to k-wise uniformity is also discussed in [2] on hardness of random CSP.

Alon et al. [3] investigated the Closeness Problem further by improving the upper bound to $O((n \log n)^{k/2}\epsilon)$. Indeed, they showed a strictly stronger fact that a distribution is $O\left(\sqrt{\mathbf{W}^{1\dots k}[\varphi]} \log^{k/2} n\right)$-close to some k-wise uniform, where $\mathbf{W}^{1\dots k}[\varphi] = \sum_{1\le|S|\le k} \widehat{\varphi}(S)^2$. Rubinfeld and Xie [27] generalized some of these results to non-uniform k-wise independent distributions over larger product spaces.

Let us briefly summarize the method [3] used to prove their upper bounds. Given an (ϵ, k)-wise uniform φ, They first try to generate a k-wise uniform "pseudo-distribution" φ' by forcing all Fourier coefficients at degree at most k to be zero. It is a "pseudo-distribution" because some points might have negative density. After this, they use a fully uniform distribution and k-wise uniform distributions with small support size to try to mend all points to be non-negative. They bound the weight of these mending distributions to upper-bound the distance incurred by mending process. This mending process uses the fully uniform distribution to mend the small negative weights and uses k-wise uniform distributions with small support size to correct the large negative weights point by point. By optimizing the threshold between small and large weights it introduces a factor of $(\log n)^{k/2}$.

Though they did not mention it explicitly, they also give a lower bound for the Closeness Problem of $\delta \ge \Omega\left(\frac{n^{(k-1)/2}}{\log n}\epsilon\right)$ for $k > 2$ by considering the uniform distribution on a set of $O(n^k)$ random chosen strings. No previous work gave any lower bound for the most natural case of $k = 2$.

1.2.2 Our result

In this paper, we show sharper upper and lower bounds for the Closeness Problem, which are tight for k even and $k = 1$. Comparing to the result in [3], we get rid of the factor of $(\log n)^{k/2}$.

▶ **Theorem 1.** *Any density φ over $\{-1, 1\}^n$ is δ-close to some k-wise uniform distribution, where*

$$\delta \le e^k \sqrt{\mathbf{W}^{1\dots k}[\varphi]} = e^k \sqrt{\sum_{1\le|S|\le k} \widehat{\varphi}(S)^2}.$$

Consequently, if φ is (ϵ, k)-wise uniform, i.e., $|\widehat{\varphi}(S)| \le \epsilon$ for every non-empty set S with size at most k, then

$$\delta \le e^k n^{k/2} \epsilon.$$

For the special case $k = 1$, the corresponding δ can be further improved to $\delta \le \epsilon$.

Our new technique is trying to mend the original distribution to be k-wise uniform all at once. We want to show that some mixture distribution $(\varphi + w\psi)$ is k-wise uniform with small mixture weight w. The distance between the final mixture distribution and the original distribution φ is bounded by $O(w)$. Therefore we only need to show that the mending distribution ψ exists for some small weight w. Showing the existence of such a distribution ψ can be written as the feasibility of a linear program (LP). We upper bound w by bounding the dual LP, using the hypercontractivity inequality.

Our result is sharp for all even k, and is also sharp for $k = 1$. We state the matching lower bound for even k:

▶ **Theorem 2.** *For any n and even k, and small enough ϵ, there exists some (ϵ, k)-wise uniform distribution φ over $\{-1, 1\}^n$, such that φ is δ-far from every k-wise uniform distribution in total variation distance, where*

$$\delta \geq \Omega\left(\frac{1}{k}\right)^k n^{k/2}\epsilon.$$

Our method for proving this lower bound is again LP duality. Our examples in the lower bound are symmetric distributions with Fourier weight only on level k. The density functions then can be written as binary Krawtchouk polynomials which behave similar to Hermite polynomials when n is large. Our dual LP bounds use various properties of Krawtchouk and Hermite polynomials.

Interestingly both our upper and lower bound utilize LP-duality, which we believe is the most natural way of looking at this problem.

We remark that we can derive a lower bound for odd k from Theorem 2 trivially by replacing k by $k-1$. There exists a gap of $\sqrt{n}$ between the resulting upper and lower bounds for odd k. We believe that the lower bound is tight, and the upper bound may be improvable by a factor of $\sqrt{n}$, as it is in the special case $k = 1$. We leave it as a conjecture for further work:

▶ **Conjecture 3.** *Suppose the distribution φ over $\{-1, 1\}^n$ is (ϵ, k)-wise uniform. Then φ is δ-close to some k-wise uniform distribution in total variation distance, where*

$$\delta \leq O(n^{\lfloor k/2 \rfloor}\epsilon).$$

1.3 The Testing Problem

Another application of the Closeness Problem is to property testing of k-wise uniformity. Suppose we have sample access from an unknown and arbitrary distribution; we may wonder whether the distribution has a certain property. This question has received tremendous attention in the field of statistics. The main goal in the study of property testing is to design algorithms that use as few samples as possible, and to establish lower bound matching these sample-efficient algorithms. In particular, we consider the property of being k-wise uniform:

Given sample access to an unknown and arbitrary distribution φ on $\{-1, 1\}^n$, how many samples do we need to distinguish between the case that φ is k-wise uniform versus the case that φ is δ-far from every k-wise uniform distribution?

In this paper, we will refer to this question as *the Testing Problem*.

We say a testing algorithm is *a δ-tester for k-wise uniformity* if the algorithm outputs "Yes" with high probability when the distribution φ is k-wise uniform, and the algorithm outputs "No" with high probability when the distribution φ is δ-far from any k-wise uniform distribution (in total variation distance).

Property testing is well studied for Boolean functions and distributions. Previous work studied testing related properties of distribution, including uniformity [16, 9, 26] and independence [8, 10, 1, 15].

[6, 3, 28] discussed about testing k-wise uniformity. [6] constructed a δ-tester for k-wise uniformity with sample complexity $O(n^{2k}/\delta^2)$, and [3] improved it to $O(n^k \log^{k+1} n/\delta^2)$. As for lower bounds, [3] show that $\Omega(n^{(k-1)/2}/\delta)$ samples are necessary, albeit only for $k > 2$. This lower bound is in particular for distinguishing the uniform distribution from δ-far-from-k-wise.

We show a better upper bound for sample complexity:

▶ **Theorem 4.** *There exists a δ-tester for k-wise uniformity of distributions on $\{-1,1\}^n$ with sample complexity $O\left(\frac{1}{k}\right)^{k/2} \frac{n^k}{\delta^2}$. For the special case of $k = 1$, the sample complexity is $O\left(\frac{\log n}{\delta^2}\right)$.*

A natural δ-tester of k-wise uniformity is mentioned in [3]: Estimate all Fourier coefficients up to level k from the samples. If they are all smaller than ϵ then output "Yes". In fact this algorithm is exactly attempting to check whether the distribution is (ϵ, k)-wise uniform. Hence the sample complexity depends on the upper bound for the Closeness Problem. Therefore we can reduce the sample complexity of this algorithm down to $O\left(\frac{n^k \log n}{\delta^2}\right)$ via our improved upper bound for the Closeness Problem. One $\log n$ factor remains because we need to union-bound over the $O(n^k)$ Fourier coefficients up to level k. To further get rid of the last $\log n$ factor, we present a new algorithm that estimates the Fourier weight up to level k, $\sum_{1 \le |S| \le k} \widehat{\varphi}^2(S)$, rather than estimating these Fourier coefficients one by one.

Unfortunately, a lower bound for the Closeness Problem does not imply a lower bound for the Testing Problem directly. In [3], they showed that a uniform distribution over a random subset of $\{-1,1\}^n$ of size $O(\frac{n^{k-1}}{\delta^2})$, is almost surely δ-far from any k-wise uniform distribution. On the other hand, by the Birthday Paradox, it is hard to distinguish between the fully uniform distribution on all strings of length n and a uniform distribution over a random set of such size. This gives a lower bound for the Testing Problem as $\Omega(n^{(k-1)/2}/\delta)$. Their result only holds for $k > 2$; there was no previous non-trivial lower bound for testing pairwise uniformity. We show a lower bound for the pairwise case.

▶ **Theorem 5.** *Any δ-tester for pairwise uniformity of distributions on $\{-1,1\}^n$ needs at least $\Omega(\frac{n}{\delta^2})$ samples.*

For this lower bound we analyze a symmetric distribution with non-zero Fourier coefficients only on level 2. We prove that it is hard to distinguish a randomly shifted version of this distribution from the fully uniform distribution. This lower bound is also better than [3] in that we have a better dependence on the parameter δ ($\frac{1}{\delta^2}$ rather than $\frac{1}{\delta}$). Unfortunately we are unable to generalize our lower bound for higher k.

Notice that for our new upper and lower bounds for k-wise uniformity testing, there still remains a quadratic gap, for $k \ge 2$, indicating that the upper bound might be able to be improved. Both the lower bound in our paper and that in [3] show that it is hard to distinguish between the fully uniform distribution and some specific sets of distributions that are far from k-wise uniform. We show that if one wants to improve the lower bound, one will need to use a distribution in the "Yes" case that is *not* fully uniform, because we give a sample-efficient algorithm for distinguishing between fully uniform and δ-far from k-wise uniform:

Table 1 Summary of our results.

	Upper bound		Lower bound	
	[3]	Our paper	[3]	Our paper
Closeness Problem	$O(n^{k/2}(\log n)^{k/2}\epsilon)$	$O(n^{k/2}\epsilon)$ $O(\epsilon)$ for $k=1$	$\Omega\left(\frac{n^{(k-1)/2}}{\log n}\epsilon\right)$	$\Omega(n^{\lfloor k/2\rfloor}\epsilon)$
Testing k-wise vs. far from k-wise	$O\left(\frac{n^k(\log n)^{k+1}}{\delta^2}\right)$	$O\left(\frac{n^k}{\delta^2}\right)$ $O\left(\frac{\log n}{\delta^2}\right)$ for $k=1$	$\Omega\left(\frac{n^{(k-1)/2}}{\delta}\right)$ for $k>2$	$\Omega\left(\frac{n}{\delta^2}\right)$ for $k=2$
Testing n-wise vs. far from k-wise	$O\left(\frac{n^k(\log n)^{k+1}}{\delta^2}\right)$	$O\left(\frac{n^{k/2}}{\delta^2}(\log\frac{n}{\delta})^{k/2}\right)$ $O\left(\frac{\log n}{\delta^2}\right)$ for $k=1$	$\Omega\left(\frac{n^{(k-1)/2}}{\delta}\right)$ for $k>2$	$\Omega\left(\frac{n}{\delta^2}\right)$ for $k=2$

▶ **Theorem 6.** *For any constant k, for testing whether a distribution is fully uniform or δ-far from every k-wise uniform distribution, there exists an algorithm with high probability ($>\frac{2}{3}$) with sample complexity $O(k)^k\cdot n^{k/2}\cdot\frac{1}{\delta^2}\cdot\left(\log\frac{n}{\delta}\right)^{k/2}$.*

In fact, for testing whether a distribution is αk-wise uniform or δ-far from k-wise uniform with $\alpha>4$ (assuming αk is an even integer), there exists an algorithm with high probability ($>\frac{2}{3}$) with sample complexity $O(\alpha)^{k/2}\cdot n^{k/2}\cdot\frac{1}{\delta^2}\cdot\left(\frac{n^k}{\delta^4}\right)^{1/(\alpha-2)}$.

We remark that testing fully uniformity can be treated as a special case of testing αk-wise uniformity approximately, by setting $\alpha=\log\frac{n}{\delta}$.

Testing full uniformity has been studied in [16, 9]. Paninski [24] showed that testing whether an unknown distribution on $\{-1,1\}^n$ is $\Theta(1)$-close to fully uniform requires $2^{n/2}$ samples. Rubinfeld and Servedio [26] studied testing whether an unknown monotone distribution is fully uniform or not.

The fully uniform distribution has the nice property that every pair of samples is different in $\frac{n}{2}\pm O(\sqrt{n})$ bits with high probability when the sample size is small. Our algorithm first rejects those distributions that disobey this property. We show that the remaining distributions have small Fourier weight up to level $2k$. Hence by following a similar analysis as the tester in Theorem 4, we can get an improved upper bound when these lower Fourier weights are small.

The lower bound remains the same as testing k-wise vs. far from k-wise. Our tester is tight up to a logarithm factor for the pairwise case, and is tight up to a factor of $\tilde{O}(\sqrt{n})$ when $k>2$.

We compare our result and previous best known bounds from [3] in Table 1. (We omit factors depending on k.)

1.4 Organization

We keep our notations the same as in [23], or see the detailed preliminaries in the full version. We will discuss upper and lower bounds for the Closeness Problem in Section 2. We will discuss the sample complexity of testing k-wise uniformity in Section 3. We present a tester for distinguishing between αk-wise uniformity (or fully uniformity) and far-from k-wise uniformity in Section 4.

2 The Closeness Problem

In this section, we prove the upper bound in Theorem 1 and the lower bound in Theorem 2. One interesting fact is that we use duality of linear programming (LP) in both the upper and lower bound. We think this is the proper perspective for analyzing these questions.

2.1 Upper bound

The key idea for proving the upper bound is mixture distributions. Given an (ϵ, k)-wise uniform density φ, we try to mix it with some other distribution ψ using mixture weight w, such that the mixture distribution $\frac{1}{1+w}(\varphi + w\psi)$ is k-wise uniform and is close to the original distribution. The following lemma shows that the distance between the original distribution and the mixture distribution is bounded by the weight w.

▶ **Lemma 7.** *If $\varphi' = \frac{1}{1+w}(\varphi + w\psi)$ for some $0 \le w \le 1$ and density functions φ, ψ, then $d_{\mathrm{TV}}(\varphi, \varphi') \le w$.*

Proof. $d_{\mathrm{TV}}(\varphi, \varphi') = \frac{1}{2}\|\varphi' - \varphi\|_1 = \frac{1}{2}\|\varphi' - ((1+w)\varphi' - w\psi))\|_1 = \frac{1}{2}w\|\varphi' - \psi\|_1 \le w.$ ◀

Therefore we only need to show the existence of an appropriate ψ for some small w. The constraints on ψ can be written as an LP feasibility problem. Therefore by Farkas' Lemma we only need to show that its dual is not feasible. The variables in the dual LP can be seen as a density function of degree at most k.

Proof of Theorem 1 (general k case). Given density function φ, we try to find another density function ψ with constraints

$$\widehat{\psi}(S) = -\frac{1}{w}\widehat{\varphi}(S)$$

for all $1 \le |S| \le k$. Suppose such a density function ψ exists. Then it is trivial that $\frac{\varphi + w\psi}{1+w}$ is also a density function and is k-wise uniform. By Lemma 7, we know that $d_{\mathrm{TV}}(\varphi, \mathrm{kWISE}) \le w$.

The rest of proof is to show that such a ψ exists when $w = e^k\sqrt{\mathbf{W}^{1\dots k}[\varphi]}$. We can write it as an LP with variables $\psi(x)$ for $x \in \{-1,1\}^n$ and constraints:

$$\begin{aligned}
&\widehat{\psi}(\emptyset) = 1, \\
&\widehat{\psi}(S) = -\frac{1}{w}\widehat{\varphi}(S), && \forall 1 \le |S| \le k, \\
&\psi(x) \ge 0, && \forall x \in \{-1,1\}^n,
\end{aligned}$$

where $\widehat{\psi}(S) = \mathbf{E}[\psi(\boldsymbol{x})\boldsymbol{x}^S]$ is a linear combination of variables $\psi(x)$.

The dual LP has variables $\psi'(x)$ for $x \in \{-1,1\}^n$ with constraints:

$$\begin{aligned}
&\widehat{\psi'}(\emptyset) = 1, \\
&\widehat{\psi'}(S) = 0, && \forall |S| > k, \\
&\psi'(x) \ge 0, && \forall x \in \{-1,1\}^n, \\
&\frac{1}{w}\sum_{1 \le |S| \le k} \widehat{\varphi}(S)\widehat{\psi'}(S) > 1.
\end{aligned}$$

The original LP is feasible if and only if its dual LP is infeasible, by Farkas' Lemma. This completes the proof, since when $w = e^k\sqrt{\mathbf{W}^{1\dots k}[\varphi]}$, for any density function ψ' with degree k we have

$$\frac{1}{w}\sum_{1 \le |S| \le k} \widehat{\varphi}(S)\widehat{\psi'}(S) \le \frac{1}{e^k\sqrt{\mathbf{W}^{1\dots k}[\varphi]}} \sum_{1 \le |S| \le k} |\widehat{\varphi}(S)||\widehat{\psi'}(S)| \le \frac{1}{e^k}\|\widehat{\psi'}\|_2 \le 1,$$

where the second inequality holds by Cauchy-Schwarz, and the last inequality holds by hypercontractivity. ◀

The proof of special case $k = 1$ for Theorem 1 is included in the appendix.

2.2 Lower bound

Interestingly, our proof of the lower bound also utilizes LP duality. We can write the closeness problem in the form of linear programming with variables $\varphi'(x)$ for $x \in \{-1,1\}^n$, as follows:

$$\begin{aligned}
\text{minimize} \quad & d_{\mathrm{TV}}(\varphi,\varphi') = \frac{1}{2}\|\varphi-\varphi'\|_1 \\
\text{subject to:} \quad & \widehat{\varphi'}(\emptyset) = 1, \\
& \widehat{\varphi'}(S) = 0, && \forall 1 \le |S| \le k, \\
& \varphi'(x) \ge 0, && \forall x \in \{-1,1\}^n.
\end{aligned}$$

We ignore the factor of $1/2$ in the minimization for convenience in the following analysis. The dual LP, which has variables $p(x), q(x)$ for $x \in \{-1,1\}^n$, is the following:

$$\begin{aligned}
\text{maximize} \quad & \langle \varphi, q\rangle - \widehat{p}(\emptyset) \\
\text{subject to:} \quad & p(x) - q(x) \ge 0, && \forall x \in \{-1,1\}^n, \\
& q(x) \le 1, && \forall x \in \{-1,1\}^n, \\
& p(x) \ge -1, && \forall x \in \{-1,1\}^n, \\
& \deg(p) \le k.
\end{aligned}$$

Thus given a pair of Boolean functions p, q satisfying the constraints, the quantity $(\langle \varphi, q\rangle - \widehat{p}(\emptyset))$ is a lower bound for our closeness problem. Our distribution φ achieving the lower bound is a symmetric polynomial, homogeneous of degree k (except that it has a constant term of 1, as is necessary for every density function). We define

$$\varphi(x) = 1 + \mu\binom{n}{k}^{-1/2} \sum_{|S|=k} x^S, \quad p(x) = \mu\binom{n}{k}^{-1/2} \sum_{|S|=k} x^S, \quad q(x) = \min(p(x), 1),$$

where $\mu = \frac{\sqrt{k!}}{2(Ck)^k}$ with some sufficiently large constant C. We have $\epsilon = \max_{1\le|S|\le k} |\widehat{\varphi}(S)| = \mu\binom{n}{k}^{-1/2}$.

We use properties of Krawtchouk and Hermite polynomials to get a lower bound of $\Omega\left(\frac{1}{k}\right)^k$ for the optimization.

This completes the proof, because φ is at least δ-far from k-wise uniform with $\delta = \Omega\left(\frac{1}{k}\right)^k$, and we have $\epsilon = \mu\binom{n}{k}^{-1/2} \le \frac{n^{-k/2}}{2^{\Omega(k)}}$. Therefore we have $\delta \ge \Omega\left(\frac{1}{k}\right)^k n^{k/2}\epsilon$.

The detailed proof of Theorem 2 is included in the full version.

3 The Testing Problem

In this section, we study the problem of testing whether a distribution is k-wise uniform or δ-far from k-wise uniform. These bounds are based on new bounds for the Closeness Problem. We present a new testing algorithm for general k in Section 3.1. We give a lower bound for the pairwise case in Section 3.2.

3.1 Upper bound

Given m samples from φ, $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$, we will first show that

$$\Delta(\boldsymbol{X}) = \operatorname*{avg}_{1\le s<t\le m} \left(\sum_{1\le|S|\le k} \boldsymbol{x}_s^S \boldsymbol{x}_t^S \right)$$

is a natural estimator of $\mathbf{W}^{1\ldots k}[\varphi]$.

▶ **Lemma 8.**

$$\mu = \mathbf{E}[\Delta(\boldsymbol{X})] = \mathbf{W}^{1\dots k}[\varphi];$$
$$\mathbf{Var}[\Delta(\boldsymbol{X})] \le \frac{4}{m^2} L_k(\varphi) + \frac{4}{m}\sqrt{L_k(\varphi)\mu}, \tag{1}$$

where $L_k(\varphi) = \sum_{1\le|S_1|,|S_2|\le k} \widehat{\varphi}(S_1 \oplus S_2)^2$.

Hence we can bound the samples we need for estimating $\mathbf{W}^{1\dots k}$.

▶ **Theorem 9** ($\mathbf{W}^{1\dots k}$ Estimation Test)**.** *Let* $\varphi : \{-1,1\}^n \to \mathbb{R}^{\ge 0}$ *be a density function, promised to satisfy* $\mathbf{W}^i[\varphi] \le An^{i/2}$ *for any* $i = 0, 1, \dots, 2k$. *There is an algorithm that, given*

$$m \ge 1000\frac{2^k\sqrt{A}n^{k/2}}{\theta} \tag{2}$$

samples, distinguishing with probability at least $3/4$ *whether* $\mathbf{W}^{1\dots k}[\varphi] \le \frac{1}{2}\theta$ *or* $\mathbf{W}^{1\dots k}[\varphi] > \theta$.

This $\mathbf{W}^{1\dots k}$ Estimation Test is just what we need for testing k-wise uniformity with the upper bound of the Closeness Problem.

Proof of Theorem 4. From Theorem 1 we know that if density φ is δ-far from k-wise uniform, then $\mathbf{W}^{1\dots k}[\varphi] > \left(\frac{\delta}{e^k}\right)^2$; On the other hand if φ is k-wise uniform, by definition we have $\mathbf{W}^{1\dots k}[\varphi] = 0$. Therefore distinguishing between k-wise uniform and δ-far from k-wise uniform can be reduced to distinguishing between $\mathbf{W}^{1\dots k}[\varphi] > \left(\frac{\delta}{e^k}\right)^2$ and $\mathbf{W}^{1\dots k}[\varphi] = 0$.

For any density function φ, $|\widehat{\varphi}(S)| = \left|\mathbf{E}[\varphi(\boldsymbol{x})\boldsymbol{x}^S]\right| \le 1$ for any $S \subseteq [n]$. Therefore assigning $A = n^k$, we have

$$\mathbf{W}^i[\varphi] = \sum_{|S|=i} \widehat{\varphi}(S)^2 \le n^i \le An^{i/2}$$

for every $i = 0, 1, \dots, 2k$.

Hence we can run the $\mathbf{W}^{1\dots k}$ Estimator Test in Theorem 9 with parameter $\theta = \left(\frac{\delta}{e^k}\right)^2$ and $A = n^k$, the we solve then Testing Problem with sample complexity $2^{O(k)}n^k/\delta^2$.

In fact by precise calculation we can further improve the factor only related to k to $O\left(\frac{1}{k}\right)^{k/2}$, but we will omit the proof here. ◀

The proofs of Lemma 8 and Theorem 9 are included in the appendix.

3.2 Lower bound for the pairwise case

An upper bound for the Closeness Problem implies an upper bound for the Testing Problem. But a lower bound for Closeness does not obviously yield a lower bound for the testing problem. The function used to show the lower bound for the Closeness Problem is far from k-wise uniform, but it is not sufficient to say that it is hard to distinguish between it and some k-wise uniform distribution. In [3], they show that it is hard to distinguish between the fully uniform distribution and the uniform distribution on a random set of size around $O(n^{k-1}/\delta^2)$; this latter distribution is far from k-wise uniform with high probability for $k > 2$.

We show that the density function φ we used for the lower bound for the Closeness Problem is a useful density to use for a testing lower bound in the pairwise case. However it is not hard to distinguish between the fully uniform distribution and φ. Our trick is

giving φ a random "center". We remind the reader that we denote by $\varphi^{+t}(x) = \varphi(x \circ t)$ the distribution φ shifted by vector t. We claim that with $m = o(n/\delta^2)$ samples, it is hard to distinguish the fully uniform distribution from φ^{+t} with a uniformly randomly chosen t.

▶ **Lemma 10.** *Let φ be the density function defined by $\varphi(x) = 1 + \frac{\delta}{n}\sum_{i<j} x_i x_j$. Assume $m < n/\delta^2$. Let $\Phi : (\{-1,1\}^n)^m \to \mathbb{R}^{\geq 0}$ be the density associated to the distribution on m-tuples of strings defined as follows: First, choose t in $\{-1,1\}^n$ uniformly; then choose m strings independently from φ^{+t}. Let $\mathbf{1}$ denote the constantly 1 function on $(\{-1,1\}^n)^m$, the density associated to the uniform distribution. Then the χ^2-divergence between Φ and $\mathbf{1}$, $\|\Phi - \mathbf{1}\|_2^2$, is bounded by:*

$$\|\Phi - \mathbf{1}\|_2^2 \leq O\left(\frac{m\delta^2}{n}\right).$$

The proof of lemma 10 is included in the full paper.

Now we are ready to give the lower bound for sample complexity of testing fully uniform vs. far-from-pairwise uniform.

Proof of Theorem 5. If $m = o(n/\delta^2)$, by Lemma 10 we have $\|\Phi - \mathbf{1}\|_2^2 \leq o(1)$. Then any tester cannot distinguish, with more than $o(1)$ advantage, whether those m samples are fully uniform or they are drawn from φ^{+t} for some random t.

On the other hand, the proof of Theorem 2 shows that φ is $\Omega(\delta)$-far from pairwise uniform, and from the Fourier characterization, we have that φ^{+t} is pairwise uniform whenever φ is. We can conclude that testing fully uniform versus δ-far from pairwise-uniform needs sample complexity at least $\Omega(n/\delta^2)$. ◀

Unfortunately, we do not see an obvious way to generalize this lower bound to $k > 2$.

4 Testing αk-wise/fully uniform vs. far from k-wise uniform

4.1 The algorithm

In this section we show a sample-efficient algorithm for testing whether a distribution is αk-wise/fully uniform or δ-far from k-wise uniform. As a reminder, Theorem 9 indicates that the sample complexity of estimating $\mathbf{W}^{1\dots k}[\varphi]$ is bounded by the Fourier weight up to level $2k$. This suggests using a filter test to try to "kick out" those distributions with noticeable Fourier weight up to degree $2k$.

Filter Test. Draw m_1 samples from φ. If there exists a pair of samples $\boldsymbol{x}, \boldsymbol{y}$ such that $|\sum_{i=1}^n \boldsymbol{x}_i \boldsymbol{y}_i| > t\sqrt{n}$, output "Reject"; Otherwise, output "Accept".

The full algorithm is combining the Filter Test and $\mathbf{W}^{1\dots k}$ Estimation Test.

Full Algorithm. Do Filter Test with m_1 samples and parameter t. If it rejects, say "No". Otherwise, do $\mathbf{W}^{1\dots k}$ Estimation Test with m_2 samples and $\theta = (\delta/e^k)^2$. Say "No" if it outputs "$\mathbf{W}^{1\dots k}[\varphi] > \theta$" and say "Yes" otherwise.

Here "Yes" means φ is αk/fully uniform, and "No" means φ is δ-far from k-wise uniform. We will decide the parameters m_1, t, m_2 in the Filter Test and the full algorithm later.

For simplicity, we define $\overline{k} = \alpha k$. We will focus on testing αk-wise uniform vs. far from k-wise uniform in the analysis. For fully uniformity, the analysis is almost the same, and we will discuss it at the end of this subsection.

First of all, we show that if φ is $\overline{k}$-wise uniform, it will pass the Filter Test with high probability, when we choose m_1 and t properly.

▶ **Lemma 11.** *If φ is $\overline{k}$-wise uniform (assuming $\overline{k}$ is even), the Filter Test will accept with probability at least .9 when $m_1^2 \leq \frac{t^{\overline{k}}}{5\overline{k}^{\overline{k}/2}}$.*

Secondly, we claim that for any distribution φ that does not get rejected by the Filter Test, it is close to a distribution φ' with upper bounds on Fourier weights of each of its levels.

▶ **Lemma 12.** *Any distribution φ either gets rejected by the Filter Test with probability at least .9, or there exists some distribution φ' such that:*

1. *φ' and φ are $\frac{8}{m_1}$-close in total variation distance;*
2. *$\mathbf{W}^i[\varphi'] \leq \frac{10^7}{m_1^2} n^i + t^i n^{i/2}$ for all $i = 1, \ldots, n$.*

If φ is not rejected by the Filter Test, Lemma 12 tells us that it is close to some distribution φ' with bounded Fourier weights on each of its levels. Even though we are drawing samples from φ, we can "pretend" that we are drawing samples from φ' since the are close.

▶ **Claim 13.** *Let $m_2 \leq \frac{m_1}{200}$, and let $A(X^{(m_2)})$ be any event related to m_2 variables in $\{-1, 1\}^n$, $X^{(m_2)} = \{x_1, \ldots, x_{m_2}\}$. Then we have*

$$\left| \Pr_{\boldsymbol{X}^{(m_2)} \sim \varphi} [A(\boldsymbol{X}^{(m_2)})] - \Pr_{\boldsymbol{X}^{(m_2)} \sim \varphi'} [A(\boldsymbol{X}^{(m_2)})] \right| \leq .08,$$

when φ and φ' are $\frac{8}{m_1}$-close.

Now we are ready to analyze the full algorithm.

We set the parameter $t = \left(10^{11}(4e^4)^k \overline{k}^{\overline{k}/2} \frac{n^k}{\delta^4}\right)^{\frac{1}{\overline{k}-2k}}$ and $m_1 = \sqrt{\frac{t^{\overline{k}}}{5\overline{k}^{\overline{k}/2}}}$ in the Filter Test; and set $m_2 = \frac{1}{200} m_1$ and $\theta = \left(\frac{\delta}{e^k}\right)^2$ in the $\mathbf{W}^{1\ldots k}$ Estimation test.

In total we use $m_1 + m_2 = O\left(\sqrt{\frac{t^{\overline{k}}}{\overline{k}^{\overline{k}/2}}}\right)$ samples in the full algorithm. By plugging in the definition of t and $\overline{k} = \alpha k$, we can simplify the sample complexity to $O(\alpha)^{k/2} \cdot n^{k/2} \cdot \frac{1}{\delta^2} \cdot \left(\frac{n^k}{\delta^4}\right)^{1/(\alpha-2)}$.

By careful calculation, we derive that the full algorithm outputs the correct answer with probability at least 2/3.

For distinguishing between a distribution of being fully uniform and a distribution of being δ-far from k-wise uniform, the modification we need is that, in Lemma 11, we use Hoeffding's inequality and get

$$\Pr_{\boldsymbol{x}, \boldsymbol{y} \sim \varphi} \left[\left| \sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{y}_i \right| > t\sqrt{n} \right] \leq 2e^{-t^2/2}$$

and then have the constraint $m_1^2 \leq \frac{1}{10} e^{t^2/2}$. Following exactly the same analysis, we get the same algorithm with sample complexity $O(k)^k \cdot n^{k/2} \cdot \frac{1}{\delta^2} \cdot \left(\log \frac{n}{\delta}\right)^{k/2}$.

The proofs of Lemma 11, Lemma 12, Claim 13 and Theorem 6 are in the appendix. The proof of Lemma 12 is included in the full version.

References

1 Jayadev Acharya, Constantinos Daskalakis, and Gautam Kamath. Optimal testing for properties of distributions. In *Advances in Neural Information Processing Systems*, pages 3591–3599, 2015.

2 Sarah R. Allen, Ryan O'Donnell, and David Witmer. How to refute a random CSP. In *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science*, pages 689–708, 2015.

3 Noga Alon, Alexandr Andoni, Tali Kaufman, Kevin Matulef, Ronitt Rubinfeld, and Ning Xie. Testing k-wise and almost k-wise independence. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing*, pages 496–505, 2007.

4 Noga Alon, László Babai, and Alon Itai. A fast and simple randomized parallel algorithm for the maximal independent set problem. *Journal of Algorithms*, 7(4):567–583, 1986.

5 Noga Alon, Oded Goldreich, Johan Håstad, and René Peralta. Simple constructions of almost k-wise independent random variables. *Random Structures & Algorithms*, 3(3):289–304, 1992.

6 Noga Alon, Oded Goldreich, and Yishay Mansour. Almost k-wise independence versus k-wise independence. *Information Processing Letters*, 88(3):107–110, 2003. doi:10.1016/S0020-0190(03)00359-4.

7 Per Austrin and Elchanan Mossel. Approximation resistant predicates from pairwise independence. *Computational Complexity*, 18(2):249–271, 2009. doi:10.1007/s00037-009-0272-6.

8 Tuğkan Batu, Eldar Fischer, Lance Fortnow, Ravi Kumar, Ronitt Rubinfeld, and Patrick White. Testing random variables for independence and identity. In *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science*, pages 442–451, 2001. doi:10.1109/SFCS.2001.959920.

9 Tuğkan Batu, Lance Fortnow, Ronitt Rubinfeld, Warren D. Smith, and Patrick White. Testing that distributions are close. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, pages 259–269, 2000. doi:10.1109/SFCS.2000.892113.

10 Tuğkan Batu, Ravi Kumar, and Ronitt Rubinfeld. Sublinear algorithms for testing monotone and unimodal distributions. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing, Chicago, IL, USA, June 13-16, 2004*, pages 381–390, 2004. doi:10.1145/1007352.1007414.

11 Louay M. J. Bazzi. Polylogarithmic independence can fool DNF formulas. *SIAM Journal on Computing*, 38(6):2220–2272, 2009. doi:10.1137/070691954.

12 Mark Braverman. Polylogarithmic independence fools AC^0 circuits. *Journal of the ACM*, 57(5):28:1–28:10, 2010. doi:10.1145/1754399.1754401.

13 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*, pages 670–683, 2016. doi:10.1145/2897518.2897528.

14 Benny Chor, Oded Goldreich, Johan Håstad, Joel Friedman, Steven Rudich, and Roman Smolensky. The bit extraction problem of t-resilient functions. In *Proceedings of the 26th Annual Symposium on Foundations of Computer Science*, pages 396–407, 1985. doi:10.1109/SFCS.1985.55.

15 Ilias Diakonikolas and Daniel M Kane. A new approach for testing properties of discrete distributions. In *Proceedings of the 57th Annual IEEE Symposium on Foundations of Computer Science*, pages 685–694. IEEE, 2016.

16 Oded Goldreich and Dana Ron. On testing expansion in bounded-degree graphs. In *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation*, pages 68–75. Springer, 2011.

17 Richard M. Karp and Avi Wigderson. A fast parallel algorithm for the maximal independent set problem. *Journal of the ACM*, 32(4):762–773, 1985. doi:10.1145/4221.4226.

18 Pravesh K. Kothari, Ryuhei Mori, Ryan O'Donnell, and David Witmer. Sum of squares lower bounds for refuting any CSP. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 132–145, 2017. doi:10.1145/3055399.3055485.

19 Xin Li. Improved two-source extractors, and affine extractors for polylogarithmic entropy. In *Proceedings of the 57th Annual IEEE Symposium on Foundations of Computer Science*, pages 168–177. IEEE, 2016.

20 Michael Luby. A simple parallel algorithm for the maximal independent set problem. *SIAM Journal on Computing*, 15(4):1036–1053, 1986. doi:10.1137/0215074.

21 Florence Jessie MacWilliams and Neil James Alexander Sloane. *The theory of error-correcting codes*. Elsevier, 1977.

22 Joseph Naor and Moni Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM Journal on Computing*, 22(4):838–856, 1993. doi:10.1137/0222053.

23 Ryan O'Donnell. *Analysis of Boolean functions*. Cambridge University Press, 2014.

24 Liam Paninski. A coincidence-based test for uniformity given very sparsely sampled discrete data. *IEEE Transactions on Information Theory*, 54(10):4750–4755, 2008. doi:10.1109/TIT.2008.928987.

25 Calyampudi Radhakrishna Rao. Factorial experiments derivable from combinatorial arrangements of arrays. *Journal of the Royal Statistical Society*, 9(1):128–139, 1947.

26 Ronitt Rubinfeld and Rocco A. Servedio. Testing monotone high-dimensional distributions. *Random Structures & Algorithms*, 34(1):24–44, 2009. doi:10.1002/rsa.20247.

27 Ronitt Rubinfeld and Ning Xie. Robust characterizations of k-wise independence over product spaces and related testing results. *Random Structures & Algorithms*, 43(3):265–312, 2013.

28 Ning Xie. *Testing k-wise independent distributions*. PhD thesis, Massachusetts Institute of Technology, 2012.

A Proof omitted from Section 2

Proof of Theorem 1 (case $k = 1$). By identifying each x_i with $-x_i$ if necessary, we may assume without loss of generality that $\widehat{\varphi}(\{i\}) \geq 0$ for all i. In addition, by reordering the coordinates, we may assume without loss of generality that $0 \leq \widehat{\varphi}(\{1\}) \leq \cdots \leq \widehat{\varphi}(\{n\}) = \epsilon$. Define ψ_j to be the density of the distribution over $\{-1,1\}^n$ which is uniform on coordinates $x_1, \ldots, x_{j-1}$, and has x_i constantly fixed to be -1 for $j \leq i \leq n$. It is easy to check $\widehat{\psi_j}(\{i\}) = 0$ for $i < j$ and $\widehat{\psi_j}(\{i\}) = -1$ for $i \geq j$.

We define φ' as

$$\varphi' = \frac{1}{1+\epsilon}\left(\varphi + \sum_{j=1}^{n} w_j \psi_j\right),$$

where

$$w_1 = \widehat{\varphi}(\{1\}), \qquad w_j = \widehat{\varphi}(\{j\}) - \widehat{\varphi}(\{j-1\}) \quad \forall 1 < j \leq n.$$

It is easy to check that φ' is a density function and

$$\widehat{\varphi'}(\{i\}) = \frac{1}{1+\epsilon}\left(\widehat{\varphi}(\{i\}) + \left(\sum_{j=1}^{i} w_j\right)(-1)\right) = 0.$$

Therefore φ' is 1-wise uniform. Then by Lemma 7,

$$d_{TV}(\varphi, \text{1WISE}) \le \frac{1}{2}\|\varphi - \varphi'\|_1 \le \sum_{j=1}^{n} w_j = \epsilon. \qquad \blacktriangleleft$$

B Proof omitted from Section 3

Proof of Lemma 8. We denote $F(x,y) = \sum_{1\le|S|\le k} x^S y^S$. We know that

$$\mathop{\mathbf{E}}_{\boldsymbol{x},\boldsymbol{y}\sim\varphi}[\boldsymbol{x}^S\boldsymbol{y}^S] = \mathop{\mathbf{E}}_{\boldsymbol{x}\sim\varphi}[\boldsymbol{x}^S]\mathop{\mathbf{E}}_{\boldsymbol{y}\sim\varphi}[\boldsymbol{y}^S] = \widehat{\varphi}(S)^2,$$

when $\boldsymbol{x}$ and $\boldsymbol{y}$ are independent samples drawn from φ. Therefore by linearity of expectation, $\mathbf{E}_{\boldsymbol{x},\boldsymbol{y}\sim\varphi}[F(\boldsymbol{x},\boldsymbol{y})] = \mathbf{W}^{1\dots k}[\varphi]$, and clearly by taking the average,

$$\mu = \mathbf{E}[\Delta(\boldsymbol{X})] = \mathbf{E}[\text{avg}_{s<t}F(\boldsymbol{x}_s,\boldsymbol{x}_t)] = \text{avg}_{s<t}\,\mathbf{E}[F(\boldsymbol{x}_s,\boldsymbol{x}_t)] = \mathbf{W}^{1\dots k}[\varphi].$$

We need to expand the variance:

$$\mathbf{Var}\left[\mathop{\text{avg}}_{s<t}(F(\boldsymbol{x}_s,\boldsymbol{x}_t))\right] = \frac{1}{\binom{m}{2}^2}\sum_{\substack{s<t\\ s'<t'}}\mathbf{Cov}[F(\boldsymbol{x}_s,\boldsymbol{x}_t),F(\boldsymbol{x}_{s'},\boldsymbol{x}_{t'})]. \tag{3}$$

We will discuss these covariances in three cases.

Case 1: $|\{s,t\}\cap\{s',t'\}| = 2$. Let $\boldsymbol{x},\boldsymbol{y}\sim\varphi$ be independent random variables.

$$\mathbf{Cov}[F(\boldsymbol{x},\boldsymbol{y}),F(\boldsymbol{x},\boldsymbol{y})] = \mathop{\mathbf{Var}}_{\boldsymbol{x},\boldsymbol{y}\sim\varphi}[F(\boldsymbol{x},\boldsymbol{y})] \le \mathop{\mathbf{E}}_{\boldsymbol{x},\boldsymbol{y}\sim\varphi}[F(\boldsymbol{x},\boldsymbol{y})^2] = \mathop{\mathbf{E}}_{\boldsymbol{x},\boldsymbol{y}\sim\varphi}\left[\left(\sum_{1\le|S|\le k}\boldsymbol{x}^S\boldsymbol{y}^S\right)^2\right].$$

Notice here all $\boldsymbol{x}_i$'s are Rademacher variables with $\boldsymbol{x}_i^2 = 1$, and similarly for the $\boldsymbol{y}_i$'s. Therefore

$$\begin{aligned}\mathop{\mathbf{E}}_{\boldsymbol{x},\boldsymbol{y}\sim\varphi}\left[\left(\sum_{1\le|S|\le k}\boldsymbol{x}^S\boldsymbol{y}^S\right)^2\right] &= \sum_{1\le|S_1|,|S_2|\le k}\mathop{\mathbf{E}}_{\boldsymbol{x},\boldsymbol{y}\sim\varphi}\left[\boldsymbol{x}^{S_1\oplus S_2}\boldsymbol{y}^{S_1\oplus S_2}\right]\\ &= \sum_{1\le|S_1|,|S_2|\le k}\widehat{\varphi}(S_1\oplus S_2)^2 = L_k(\varphi).\end{aligned}$$

Case 2: $|\{s,t\} \cap \{s',t'\}| = 1$. Let $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \sim \varphi$ be independent random variables. Similar to Case 1, we have:

$$\begin{aligned}
\mathbf{Cov}[F(\boldsymbol{x},\boldsymbol{y}), F(\boldsymbol{x},\boldsymbol{z})] &\le \mathbf{E}[F(\boldsymbol{x},\boldsymbol{y})F(\boldsymbol{x},\boldsymbol{z})] \\
&= \mathbf{E}\left[\left(\sum_{1\le|S_1|\le k} \boldsymbol{x}^{S_1}\boldsymbol{y}^{S_1}\right)\left(\sum_{1\le|S_2|\le k} \boldsymbol{x}^{S_2}\boldsymbol{z}^{S_2}\right)\right] \\
&= \mathbf{E}\left[\sum_{1\le|S_1|,|S_2|\le k} \boldsymbol{x}^{S_1\oplus S_2}\boldsymbol{y}^{S_1}\boldsymbol{z}^{S_2}\right] \\
&= \sum_{1\le|S_1|,|S_2|\le k} \widehat{\varphi}(S_1\oplus S_2)\widehat{\varphi}(S_1)\widehat{\varphi}(S_2) \\
&\le \sqrt{\sum_{1\le|S_1|,|S_2|\le k} \widehat{\varphi}(S_1\oplus S_2)^2}\sqrt{\sum_{1\le|S_1|,|S_2|\le k} \widehat{\varphi}(S_1)^2\widehat{\varphi}(S_2)^2} \\
&= \sqrt{L_k(\varphi)}\mu,
\end{aligned}$$

where the inequality comes from Cauchy–Schwarz.

Case 3: $|\{s,t\} \cap \{s',t'\}| = 0$. Let $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \boldsymbol{w} \sim \varphi$ be independent random variables. Clearly $F(\boldsymbol{x},\boldsymbol{y})$ and $F(\boldsymbol{z},\boldsymbol{w})$ are independent and therefore $\mathbf{Cov}[F(\boldsymbol{x},\boldsymbol{y}), F(\boldsymbol{z},\boldsymbol{w})] = 0$.

Plugging all these cases into eq. (3), we get

$$\begin{aligned}
\mathbf{Var}[\Delta(\boldsymbol{X})] &= \mathbf{Var}\left[\operatorname*{avg}_{s<t}(F(\boldsymbol{x}_s,\boldsymbol{x}_t)\right] \\
&= \frac{1}{\binom{m}{2}^2}\left(\binom{m}{2}L_k(\varphi) + m(m-1)(m-2)\sqrt{L_k(\varphi)}\mu\right) \\
&\le \frac{4}{m^2}L_k(\varphi) + \frac{4}{m}\sqrt{L_k(\varphi)}\mu.
\end{aligned}$$

◀

Proof of Theorem 9. The algorithm is simple: we report "$\mu \le \frac{1}{2}\theta$" if $\Delta(\boldsymbol{X}) \le \frac{3}{4}\theta$ and report "$\mu > \theta$" if $\Delta(\boldsymbol{X}) > \frac{3}{4}\theta$.

Now we need to bound $L_k(\varphi)$ to bound the variance of $\Delta(\boldsymbol{X})$. For a fixed subset $|S| \le 2k$, how many pairs of $1 \le |S_1|, |S_2| \le k$ are there satisfying $S = S_1 \oplus S_2$? We denote $S_1 = S_1' \cup T$, $S_2 = S_2' \cup T$, where S_1', S_2', T are disjoint. Then $S = S_1' \cup S_2'$. For a fixed set S, there are at most $2^{|S|}$ different ways to split it into two sets S_1', S_2'. Because $\max\{S_1', S_2'\} \ge \lceil |S|/2 \rceil$ and $|S_1|, |S_2| \le k$, we have $|T| \le k - \lceil |S|/2 \rceil$. Therefore there are at most

$$\sum_{j=0}^{k-\lceil|S|/2\rceil} \binom{n-|S|}{j} \le \frac{2n^{k-\lceil|S|/2\rceil}}{(k-\lceil|S|/2\rceil)!}$$

ways to choose the set T for any fixed S_1', S_2'. Hence,

$$\begin{aligned}
L_k(\varphi) &= \sum_{1\le|S_1|,|S_2|\le k} \widehat{\varphi}(S_1 \oplus S_2)^2 \\
&= \sum_{|S|\le 2k} \sum_{\substack{S_1'\cap S_2'=\emptyset \\ S_1'\cup S_2'=S}} \sum_{\substack{T\cap S_1'=\emptyset, T\cap S_2'=\emptyset \\ |T|+\max\{|S_1'|,|S_2'|\}\le k}} \widehat{\varphi}(S)^2 \\
&\le \sum_{|S|\le 2k} 2^{|S|} \frac{2n^{k-\lceil |S|/2\rceil}}{(k-\lceil |S|/2\rceil)!}\widehat{\varphi}(S)^2 \\
&= \sum_{i=0}^{2k} 2^i \frac{2n^{k-\lceil i/2\rceil}}{(k-\lceil i/2\rceil)!}\mathbf{W}^i[\varphi].
\end{aligned}$$

Plugging in $\mathbf{W}^i[\varphi] \le An^{i/2}$, we get

$$L_k(\varphi) \le \sum_{i=0}^{2k} 2^i \frac{2n^{k-\lceil i/2\rceil}}{(k-\lceil i/2\rceil)!}\mathbf{W}^i[\varphi] \le 2^{2k+2}An^k. \tag{4}$$

By substituting eq. (4) and eq. (2) into eq. (1), we have

$$\mathbf{Var}[\Delta(\boldsymbol{X})] \le \frac{4}{500^2}\theta^2 + \frac{4}{500}\theta\mu \le \frac{1}{64}\max\{\theta^2, \mu^2\}.$$

Then we conclude our proof by Chebyshev's inequality:

$$\begin{aligned}
\mathbf{Pr}\left[|\Delta(\boldsymbol{X}) - \mu| \le \frac{1}{4}\max\{\theta,\mu\}\right] &\ge \mathbf{Pr}\left[|\Delta(\boldsymbol{X}) - \mu| \le 2\sqrt{\mathbf{Var}[\Delta(\boldsymbol{X})]}\right] \\
&\ge 1 - \left(\frac{1}{2}\right)^2 = \frac{3}{4}.
\end{aligned}$$

◀

C Proofs omitted in Section 4

Proof of Lemma 11. If φ is $\overline{k}$-wise uniform with $\overline{k}$ even, then by Markov's inequality on the $\overline{k}$-th moment, we have

$$\mathop{\mathbf{Pr}}_{\substack{\boldsymbol{x},\boldsymbol{y}\sim\varphi \\ \text{independent}}}\left[\left|\sum_{i=1}^n \boldsymbol{x}_i\boldsymbol{y}_i\right| > t\sqrt{n}\right] = \mathop{\mathbf{Pr}}_{\boldsymbol{x},\boldsymbol{y}\sim\varphi}\left[\left(\sum_{i=1}^n \boldsymbol{x}_i\boldsymbol{y}_i\right)^{\overline{k}} > (t\sqrt{n})^{\overline{k}}\right] \le \frac{\mathbf{E}_{\boldsymbol{x},\boldsymbol{y}\sim\varphi}\left[\left(\sum_{i=1}^n \boldsymbol{x}_i\boldsymbol{y}_i\right)^{\overline{k}}\right]}{t^{\overline{k}}n^{\overline{k}/2}}.$$

When we expand $\left(\sum_{i=1}^n x_iy_i\right)^{\overline{k}}$, each term is at most degree $\overline{k}$ in x or y. Because $\boldsymbol{x}$ and $\boldsymbol{y}$ are independent random variables chosen from $\overline{k}$-wise uniform distribution φ, the whole polynomial behaves the same as if $\boldsymbol{x}$ and $\boldsymbol{y}$ were chosen from the fully uniform distribution:

$$\begin{aligned}
\mathop{\mathbf{E}}_{\boldsymbol{x},\boldsymbol{y}\sim\varphi}\left[\left(\sum_{i=1}^n \boldsymbol{x}_i\boldsymbol{y}_i\right)^{\overline{k}}\right] &= \mathop{\mathbf{E}}_{\boldsymbol{z}\sim\{-1,1\}^n}\left[\left(\sum_{i=1}^n \boldsymbol{z}_i\right)^{\overline{k}}\right] \\
&\le \overline{k}^{\overline{k}/2}\left(\mathop{\mathbf{E}}_{\boldsymbol{z}\sim\{-1,1\}^n}\left[\left(\sum_{i=1}^n \boldsymbol{z}_i\right)^2\right]\right)^{\overline{k}/2} \\
&= \overline{k}^{\overline{k}/2}n^{\overline{k}/2}.
\end{aligned}$$

The inequality uses hypercontractivity; see Theorem 9.21 in [23]. Hence we have

$$\Pr_{\boldsymbol{x},\boldsymbol{y}\sim\varphi}\left[\left|\sum_{i=1}^{n}\boldsymbol{x}_i\boldsymbol{y}_i\right| > t\sqrt{n}\right] \leq \frac{\overline{k}^{\overline{k}/2}}{t^{\overline{k}}}.$$

When drawing m_1 examples, there are at most $\binom{m_1}{2} \leq \frac{1}{2}m_1^2$ pairs. Hence by the union bound, the probability of φ getting rejected is at most $\frac{m_1^2\overline{k}^{\overline{k}/2}}{2t^{\overline{k}}} \leq \frac{1}{10}$. ◀

Proof of Claim 13. We denote by Φ(respectively, Φ') the joint distribution of m_2 samples from φ(respectively, φ'). Then by a union bound we know that Φ and Φ' are .04-close, since $m_2\frac{8}{m_1} \leq .04$. We denote $\mathbf{1}[A(\boldsymbol{X}^{(m_2)})]$ as the indicator function of event A happening on $\boldsymbol{X}^{(m_2)}$. Then we have

$$\begin{aligned}\left|\Pr_{\boldsymbol{X}^{(m_2)}\sim\varphi}[A(\boldsymbol{X}^{(m_2)})] - \Pr_{\boldsymbol{X}^{(m_2)}\sim\varphi'}[A(\boldsymbol{X}^{(m_2)})]\right| &= \left|\sum_{X^{(m_2)}}\mathbf{1}[A(X^{(m_2)})]\left(\Phi(X^{(m_2)}) - \Phi'(X^{(m_2)})\right)\right| \\ &\leq \sum_{X^{(m_2)}}\left|\Phi(X^{(m_2)}) - \Phi'(X^{(m_2)})\right| \\ &= 2d_{\mathrm{TV}}(\Phi,\Phi') \leq .08\end{aligned}$$

which completes the proof. ◀

Proof of Theorem 6. We discuss distinguishing between $\overline{k}$-wise uniform and δ-far from k-wise uniform first. In the Overall Algorithm, we set the parameters $t = \left(10^{11}(4e^4)^k\overline{k}^{\overline{k}/2}\frac{n^k}{\delta^4}\right)^{\frac{1}{\overline{k}-2k}}$ and $m_1 = \sqrt{\frac{t^{\overline{k}}}{5\overline{k}^{\overline{k}/2}}}$ in the Filter Test; and, we set $m_2 = \frac{1}{200}m_1$ and $\theta = \left(\frac{\delta}{e^k}\right)^2$ in the $\mathbf{W}^{1\ldots k}$ Estimation test.

In total we use $m_1 + m_2 = O\left(\sqrt{\frac{t^{\overline{k}}}{\overline{k}^{\overline{k}/2}}}\right)$ samples in the Overall Algorithm. By plugging in the definition of t and $\overline{k} = \alpha k$, we can simplify the sample complexity to $O(\alpha)^{k/2}\cdot n^{k/2}\cdot \frac{1}{\delta^2}\cdot\left(\frac{n^k}{\delta^4}\right)^{1/(\alpha-2)}$.

The rest of the proof is to show the correctness of this algorithm. We discuss the two cases.

"Yes" case: Suppose φ is $\overline{k}$-wise uniform. By Lemma 11 we know that φ will pass the Filter Test with probability at least .9 since $m_1^2 = \frac{t^{\overline{k}}}{5\overline{k}^{\overline{k}/2}}$.

Now φ is $\overline{k}$-wise uniform with $\overline{k} > 2k$, which means $\widehat{\varphi}(S) = 0$ for any $1 \leq |S| \leq 2k$. Therefore by setting $\delta = \left(\frac{\theta}{e^k}\right)^2$ and $A = 1$, Theorem 9 tells us that m_2 samples are large enough for $\mathbf{W}^{1\ldots k}$ Estimation Test to output "$\mathbf{W}^{1\ldots k}[\varphi] \leq \frac{1}{2}\theta$" with probability 3/4.

The overall probability of the Overall Algorithm saying "Yes" is therefore at least $.9 \times \frac{3}{4} > \frac{2}{3}$.

"No" case: Suppose φ is δ-far from k-wise uniform. Either φ gets rejected by the Filter Test with probability .9, or according to Lemma 12, we know that there exists some distribution φ' which is $\frac{8}{m_1}$-close to φ and $\mathbf{W}^i[\varphi'] \leq \frac{10^7}{m_1^2}n^i + t^i n^{i/2}$ for all $i = 1,\ldots,n$.

The second stage is slightly tricky. As described in Claim 13, at the expense of losing .08 probability, we may pretend we are drawing samples from φ' rather than φ. Notice that $m_1^2 = \frac{t^{\overline{k}}}{5\overline{k}^{\overline{k}/2}} = \omega(n^k)$. We have

$$\mathbf{W}^i[\varphi'] \le \frac{10^7}{m_1^2} n^i + t^i n^{i/2} = (1 + o(1))t^i n^{i/2} \le A n^{i/2}$$

for $i = 0, \ldots, 2k$ with parameter $A = 1.01t^{2k}$. Then plugging $A = 1.01t^{2k}$ and $\theta = \left(\frac{\delta}{e^k}\right)^2$ into Theorem 9, we know that the $\mathbf{W}^{1\cdots k}$ Estimation Test will say "$\mathbf{W}^{1\cdots k}[\varphi] > \theta$" with probability at least $\frac{3}{4}$ when φ' is δ-far from k-wise uniform, provided we have at least $1005\frac{(2e^2)^k t^k n^{k/2}}{\delta^2}$ samples. It is easy to check $m_2 = \frac{1}{200}\sqrt{\frac{t^{\overline{k}}}{5\overline{k}^{\overline{k}/2}}}$ is sufficient.

However, in the real algorithm we are drawing samples from φ rather than φ'. From Claim 13, we know that the estimator will accept with probability at least $\frac{3}{4} - .08 > \frac{2}{3}$ when φ' is δ-far from k-wise uniform. Notice that φ and φ' are $\frac{8}{m_1}$-close, where $\frac{8}{m_1} = o\left(\frac{\delta^4}{n^k}\right)$. Hence if φ is δ-far from k-wise uniform, φ' is also δ-far from k-wise uniform, which completes the proof.

Finally, for distinguishing between a distribution being fully uniform and a distribution being δ-far from k-wise uniform, the modification we need is that in Lemma 11 we use Hoeffding's inequality to get

$$\Pr_{\boldsymbol{x},\boldsymbol{y}\sim\varphi}\left[\left|\sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{y}_i\right| > t\sqrt{n}\right] \le 2e^{-t^2/2},$$

and then we have the constraint $m_1^2 \le \frac{1}{10}e^{t^2/2}$. Following exactly the same analysis, we get the same algorithm with sample complexity $O(k)^k \cdot n^{k/2} \cdot \frac{1}{\delta^2} \cdot \left(\log \frac{n}{\delta}\right)^{k/2}$. ◀

Pseudo-Derandomizing Learning and Approximation

Igor Carboni Oliveira
Department of Computer Science, University of Oxford, United Kingdom.
igor.carboni.oliveira@cs.ox.ac.uk

Rahul Santhanam
Department of Computer Science, University of Oxford, United Kingdom.
rahul.santhanam@cs.ox.ac.uk

Abstract

We continue the study of *pseudo-deterministic algorithms* initiated by Gat and Goldwasser [7]. A pseudo-deterministic algorithm is a probabilistic algorithm which produces a fixed output with high probability. We explore pseudo-determinism in the settings of *learning* and *approximation*. Our goal is to simulate known randomized algorithms in these settings by pseudo-deterministic algorithms in a generic fashion – a goal we succinctly term *pseudo-derandomization*.

Learning. In the setting of learning with membership queries, we first show that randomized learning algorithms can be derandomized (resp. pseudo-derandomized) under the standard hardness assumption that E (resp. BPE) requires large Boolean circuits. Thus, despite the fact that learning is an algorithmic task that requires interaction with an oracle, standard hardness assumptions suffice to (pseudo-)derandomize it. We also *unconditionally* pseudo-derandomize any quasi-polynomial time learning algorithm for polynomial size circuits on infinitely many input lengths in sub-exponential time.

Next, we establish a generic connection between learning and derandomization in the reverse direction, by showing that deterministic (resp. pseudo-deterministic) learning algorithms for a concept class $\mathfrak{C}$ imply hitting sets against $\mathfrak{C}$ that are computable deterministically (resp. pseudo-deterministically). In particular, this suggests a new approach to constructing hitting set generators against $\mathcal{AC}^0[p]$ circuits by giving a deterministic learning algorithm for $\mathcal{AC}^0[p]$.

Approximation. Turning to approximation, we *unconditionally* pseudo-derandomize any polytime randomized approximation scheme for integer-valued functions infinitely often in subexponential time over any samplable distribution on inputs. As a corollary, we get that the $(0,1)$-Permanent has a fully pseudo-deterministic approximation scheme running in sub-exponential time infinitely often over any samplable distribution on inputs.

Finally, we investigate the notion of *approximate canonization* of Boolean circuits. We use a connection between pseudodeterministic learning and approximate canonization to show that if BPE does not have sub-exponential size circuits infinitely often, then there is a pseudo-deterministic approximate canonizer for $\mathcal{AC}^0[p]$ computable in quasi-polynomial time.

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization

Keywords and phrases derandomization, learning, approximation, boolean circuits

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.55

Related Version A full version of the paper is available at `https://eccc.weizmann.ac.il/report/2018/122/`.

Funding This work was supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013)/ERC Grant No. 615075.

Acknowledgements We thank Chris Brzuska for bringing [3] to our attention, Roei Tell for helpful discussions, and the reviewers for comments that improved the presentation.

1 Introduction

1.1 Context and Motivation

Randomness is a powerful algorithmic resource, used widely in tasks such as cryptography, distributed computing, learning, sampling and approximation. Although it often makes algorithmic tasks more efficient, randomness comes with issues. It introduces *uncertainty* – running a randomized algorithm multiple times, we cannot always expect to get the same answer. Moreover, randomized algorithms assume access to a source of independent and unbiased random bits, and this assumption is not always justified in the physical world.

Ideally, we would like to perform any efficient randomized task almost as efficiently without using randomness at all, or while using as little randomness as possible. This is the goal of *derandomization*, which has been widely studied in complexity theory. While generic derandomization is possible in many settings under widely believed circuit lower bound hypotheses, it also *implies* circuit lower bounds that are believed to be hard to establish (cf. [17, 21, 37]). Thus, while many specific randomized tasks can be derandomized, provable generic derandomization seems out of reach with our current state of knowledge.

A few years ago, Gat and Goldwasser [7] introduced the notion of *pseudo-deterministic algorithms*, motivated by applications in cryptography and distributed computing. A pseudo-deterministic algorithm is one that, on a given input, produces a fixed output with very high probability. Thus, a pseudo-deterministic algorithm is one that *looks* deterministic to an outside observer who is computationally bounded – even if such an observer were to run the algorithm multiple times, she is likely to always get the same answer.

Pseudo-deterministic algorithms have a very desirable feature possessed by deterministic algorithms, viz. little to no uncertainty in the output. Thus it is of interest to convert randomized algorithms to equivalent pseudo-deterministic ones – we term such a conversion "pseudo-derandomization".

There are several interesting examples of pseudo-derandomization known, including finding primitive roots [13] and quadratic non-residues (cf. [7]) in prime fields, finding variable settings for polynomial identity testing [7], and finding perfect matchings in bipartite graphs in parallel [9]. These pseudo-derandomization results exploit specific properties of the known randomized algorithms for these problems. The authors introduced a generic pseudo-derandomization approach for search problems in [29], and used it to give a sub-exponential pseudo-deterministic construction of primes infinitely often. This generic approach has been explored further by [16] and [10].

One limitation of the generic approach of [29] is that it seems to work only for *search problems* whose underlying relation is decidable in P or in BPP. In particular, the approach requires the ability to test in P or in BPP whether a given sequence of random choices made by a randomized algorithm is "good" or not. There are several important settings of randomized tasks where such a test is not available. We consider two such settings in this paper: *learning* and *approximation*.

1.2 Pseudoderandomization and learning

Our learning model is that of learning with membership queries, where the accuracy of the output hypothesis is measured with respect to the uniform distribution. A pseudo-deterministic learning algorithm in this model is a randomized algorithm that, when given access to a predetermined oracle, makes a fixed set of queries and outputs a fixed output

hypothesis with high probability.[1] (The pseudo-deterministic learning model is formalized in the natural way in Section A.2.)

This setting falls outside the "search problem" paradigm for a couple of different reasons: first, the algorithmic task is not self-contained but requires interaction with an oracle, and second, the test of whether an output hypothesis is good is not precise but approximate, and again requires interaction with the oracle.[2]

Pseudo-determinism is naturally a desirable property for learning algorithms. Indeed, consider a setting where Alice and Bob run the same learning program independently on the same data but wish to co-ordinate their predictions. Pseudo-determinism of the learning algorithm enables them to co-ordinate their predictions perfectly with high probability. In an alternative scenario, suppose Alice runs the learning algorithm to generate a hypothesis, and the hypothesis gets corrupted. Alice can recover the original hypothesis with high probability just by running the learning algorithm again.

The main question we ask is: can learning algorithms be derandomized or at least pseudo-derandomized in a generic fashion? Our first result is the observation that standard pseudo-random generators suffice to derandomize learning. This is somewhat surprising because standard pseudo-random generators are designed for self-contained algorithmic tasks, while learning requires interaction with an unknown oracle.

Recall that we consider randomized algorithms that learn under the uniform distribution and have membership-query access to the unknown function.

▶ **Theorem 1** (Conditional derandomization and pseudo-derandomization of learning)**.** *Let $\mathfrak{C} \subseteq \mathsf{P/poly}$ be an arbitrary circuit class, and suppose $\mathfrak{C}(s(n))$ can be learned to any constant accuracy by a randomized algorithm running in time $t(n) \geq n$.*

- *If $\mathsf{E} = \mathsf{DTIME}[2^{O(n)}]$ requires circuits of size $2^{\Omega(n)}$ on all large input lengths, then there exists a constant $c \geq 1$ such that $\mathfrak{C}$ can be deterministically learned to any constant accuracy in time at most $O(t(n)^c)$.*
- *If $\mathsf{BPE} = \mathsf{BPTIME}[2^{O(n)}]$ requires circuits of size $2^{\Omega(n)}$ on all large input lengths, then there exists a constant $c \geq 1$ such that $\mathfrak{C}(s)$ can be pseudo-deterministically learned to any constant accuracy in time at most $O(t(n)^c)$.*

The proof of conditional derandomization in Theorem 1 works as follows. Under the assumption that E requires exponential-size Boolean circuits almost everywhere, it is a standard consequence from [26, 18, 34] that there is a pseudo-random generator G computable in time $\mathrm{poly}(t(n))$ with seed length $O(\log(t(n))$ secure against circuits of size $t(n)^3$. We simulate the randomized learning algorithm using each output of the generator G as random sequence in turn to obtain hypothesis circuits $D_1 \ldots D_{\mathrm{poly}(t(n))}$, and then output the majority of these circuits as our hypothesis. To argue that this works, we show that if the simulation fails to output a correct hypothesis, there is a distinguisher for the PRG G, contrary to our assumption. The key idea is that a distinguisher can be constructed in $t(n)^3$ size by replacing the oracle in the simulation of the learning algorithm by a circuit from the class $\mathfrak{C}$ for which the simulation fails. The proof of conditional pseudo-derandomization works similarly, but we need to use an additional idea from [29]. Details are in Section 2.[3]

[1] We stress that we allow adaptive learning algorithms, and that the "canonical" set of inputs queried by the learner can depend on the target function. We do not require the order of the queries to be fixed.

[2] Also observe that the usual way of testing a learning hypothesis by drawing a set of random examples is not pseudo-deterministic.

[3] For simplicity, we have restricted the statement of Theorem 1 to constant-accuracy learners. As explained in Section 2, from a randomized ε-accuracy learner one can get a deterministic $O(\varepsilon)$-accuracy learner by using sufficiently strong generators (see Lemma 6).

We get some interesting corollaries from Theorem 1. Under standard hardness assumptions, both Jackson's polynomial-time learning algorithm for DNFs with membership queries [19] and the recent algorithm of [4] for $\mathcal{AC}^0[p]$ can be derandomized. Note that the randomized learner of [24] for $\mathcal{AC}^0$ has already been derandomized unconditionally by Sitharam [32].[4] Sitharam's deterministic learner exploits specific properties of $\mathcal{AC}^0$ circuits, while we are interested here in generic methods to derandomize and pseudo-derandomize learning algorithms.

Theorem 1 is conditional, but it can be used to establish an *unconditional* result for pseudo-derandomizing learning. This is in contrast to generic derandomization, which can only be done conditionally given our current knowledge of circuit lower bounds.

▶ **Theorem 2** (Unconditional pseudo-derandomization of learning)**.**
If P/poly *can be learned to any constant accuracy by a randomized algorithm running in quasi-polynomial time, then for each* $\gamma > 0$*,* P/poly *can be pseudo-deterministically learned to any constant accuracy in time* $O(2^{n^\gamma})$ *for infinitely many input lengths* n*.*

The proof of Theorem 2 proceeds in two steps. In the first step, we use a result of [28] to get circuit lower bounds for BPE from a non-trivial randomized learning for P/poly. In the second step, we apply a variant of Theorem 1 to derive an infinitely-often subexponential-time pseudo-deterministic learner using the circuit lower bounds for BPE.

The assumption in Theorem 2 is very strong; indeed, under standard cryptographic assumptions, P/poly does not have non-trivial learning algorithms. However, the proof technique of Theorem 2 works in the more general setting of *self-learners*, where a self-learner is a learning algorithm for a circuit class $\mathfrak{C}$ that produces a hypothesis in $\mathfrak{C}$ and moreover can itself be implemented in $\mathfrak{C}$. Theorem 2 is just the cleanest instantiation of this proof technique, since any learner for P/poly is automatically a self-learner. (Self-learning is a phenomenon that might be of independent interest, and we refer to Section 3 for further discussion of this concept.) The more general version of Theorem 2 presented in Section 3 shows that the same result holds for any self-learnable class that contains $\mathcal{TC}^0$ and is closed under composition. (For the interested reader, we mention that threshold gates are necessary to perform hardness amplification, a technical ingredient in our proof.)

Theorem 1 applies pseudo-random generators to the setting of learning. Our next result goes in the opposite direction, showing that derandomizing or pseudo-derandomizing learning algorithms has interesting consequences in the theory of pseudo-randomness. We say that a circuit is γ-*dense* if it accepts at least a γ-fraction of strings in $\{0,1\}^n$.

▶ **Theorem 3** (Hitting sets from deterministic and pseudo-deterministic learning)**.**
Let $\mathfrak{C} = \{\mathcal{C}_n\}$ *be an arbitrary circuit class, and assume that for every* $\varepsilon > 0$*,* $\mathfrak{C}$*-circuits of size* $s(n)$ *can be deterministically learned to accuracy* ε *in time* $T(n) \geq n$*.*

Then, for every $\gamma > 0$*, there exists a hitting set generator* $G_n \colon \{0,1\}^{\log T(n)} \to \{0,1\}^n$ *computable in time* $O(T(n))$ *against the class of* γ*-dense circuits in* $\mathcal{C}_n(s(n))$*. Similarly, if* $\mathfrak{C}$ *is pseudodeterministically learnable, there exist pseudodeterministic hitting set generators against* $\mathcal{C}_n(s(n))$ *with the same parameters.*

The proof of Theorem 3 is along the lines of the argument used to prove that a deterministic black-box PIT algorithm implies a hitting set. Suppose that there exists a deterministic learner. We run the deterministic learner with oracle the identically zero function, and output the set of queries it makes as our candidate hitting set. If the set of queries is not a hitting

[4] See also [33] for related results in the context of learnability using a linear combination of parity functions.

set, then there must be a somewhat dense function f computable in $\mathfrak{C}$ for which all the queries answer 0, just as they do for the identically zero function. But by the correctness of the learning algorithm, this would mean that f can be well-approximated by the identically zero function, which contradicts the assumption that it is somewhat dense. The consequence for pseudo-deterministic learners is shown by appropriately adapting this argument.

An interesting application of Theorem 3 is to the question of whether small hitting sets exist for $\mathcal{AC}^0[p]$ circuits. Despite much effort, no hitting sets even of sub-exponential size are known for such circuits (we refer to [5] for related results and discussion). Theorem 3 suggests an approach to this question via learning. Carmosino et al. [4] recently gave a quasi-polynomial time randomized learning algorithm for $\mathcal{AC}^0[p]$ – if this algorithm could be made deterministic, we would immediately get quasi-polynomial size hitting sets for $\mathcal{AC}^0[p]$ in quasi-polynomial time! In particular, that would imply that randomized poly-size $\mathcal{AC}^0[p]$ circuits with one-sided error can be simulated by deterministic quasi-poly size circuits. Even a pseudo-derandomization of the [4] algorithm would be interesting, as this would give somewhat efficient pseudo-deterministic hitting sets against $\mathcal{AC}^0[p]$, which is also unknown.

Theorem 3 also has consequences for non-uniform circuit lower bounds that can be derived from learning algorithms. It is known that *non-trivial* learning algorithms (i.e., those running in time $2^n/n^{\omega(1)}$) for a circuit class $\mathfrak{C}$ yield lower bounds against $\mathfrak{C}$ (cf. [23, 28, 6, 15, 36]). However, different algorithms provide different types of lower bounds. For a deterministic learner, one obtains a function in E that is hard almost everywhere [23], while for randomized learners, the hard function lives in BPE and is only hard infinitely often [28]. Interestingly, it is possible to use Theorem 3 to get something stronger from non-trivial *pseudo-deterministic* (randomized) learning algorithms: they can be used to define a function in BPE that is hard almost-everywhere for $\mathfrak{C}$.

1.3 Pseudoderandomization and approximation

We next turn our attention to a different setting, the setting of *approximation*. We are interested in integer-valued functions, i.e., functions from strings to non-negative integers, that have efficient randomized approximation schemes. The question is whether the existence of a good randomized approximation scheme generically implies the existence of a somewhat efficient pseudo-deterministic approximation scheme. (Pseudo-deterministic approximation schemes are formalized in the natural way in Section A.3.)

Note that this setting too does not conform to the "search problem" paradigm. Given a value w, it might be hard to test if the value is close to the correct value, since the correct value might be very hard to compute. Indeed, in our results, we make no assumptions about the complexity of exact computation of the integer-valued function.

Our main result here is a generic pseudo-derandomization of randomized approximation schemes; however, this pseudo-derandomization is only guaranteed to work on infinitely many input lengths with high probability over any poly-time samplable distribution of inputs.

▶ **Theorem 4** (Unconditional pseudo-derandomization of approximation). *Let $f\colon \{0,1\}^* \to \mathbb{N}$ be any function with a polynomial-time randomized approximation scheme. Then for each polynomial-time samplable sequence $\mathfrak{D}$ of distributions and for each constant $\delta > 0$, f has a pseudo-deterministic approximation scheme for infinitely many n over $\mathfrak{D}$ running in time $O(2^{n^\delta})$.*

The main idea in the proof of Theorem 4 is to exploit the uniform hardness-randomness tradeoffs used in the generic pseudo-derandomization results of [29], but adapted to this new setting. The crucial point is: how do we test efficiently that a value w is a good approximation

to the correct value? We test this simply by running the randomized approximation scheme to produce a value w' and checking if w is close to w'. This is not a deterministic polynomial-time test or indeed a bounded-error probabilistic polynomial-time test; however, we can show that it is good enough for our purposes.

As a corollary of this result and [20], we get unconditionally that the $(0,1)$-Permanent has a pseudo-deterministic approximation scheme running in sub-exponential time on infinitely many input lengths over any poly-time samplable distribution on inputs.

Finally, we consider a notion of *approximate canonization* of circuits. Canonization is a natural notion for an equivalence relation, where for each element of the set we compute a representative member of its equivalence class. Needless to say, canonization and canonical forms are fundamental notions with a variety of applications both in mathematics and computer science. We are interested in the natural equivalence relation between circuits: two circuits are equivalent if they compute the same function.

It is not hard to prove that efficient canonization is impossible for even weak circuit classes such as DNFs, under standard complexity assumptions. Therefore we *relax* the notion of canonization. We still require the output of the canonizer to be the same for any two equivalent circuits, but this output need not be a circuit equivalent to the original circuit, instead it is allowed to be *close* to the original circuit over the uniform distribution on inputs.[5]

Inspired by an observation in [1], we show that efficient deterministic (resp. pseudo-deterministic) learning implies efficient deterministic (resp. pseudo-deterministic) approximate canonization. (We refer to Section A.3 for a precise definition of approximate canonization.) Using Theorem 1 and the learning algorithm in [4], we get quasi-polynomial time pseudo-deterministic approximate canonization for $\mathcal{AC}^0[p]$ circuits under a standard circuit lower bound assumption for BPE.

▶ **Theorem 5** (Approximate canonization for $\mathcal{AC}^0[p]$, Informal)**.**
Let $p \geq 2$ be a fixed prime. If BPE *requires circuits of size $2^{n^{\Omega(1)}}$ almost everywhere, then $\mathcal{AC}^0[p]$ circuits can be approximately canonized in pseudo-deterministic quasi-polynomial time.*

We leave as an open problem obtaining an unconditional version of this theorem. Another interesting research direction is the investigation of connections between approximate canonization and other meta-computational problems. In this sense, we mention that [3] provides evidence that expressive circuit classes do not admit approximate canonization. (In fact, even more relaxed notions of approximate canonization are conditionally ruled out by the results from [3], and we refer to their work for further details.)

1.4 Organization

We formally define our models and state some auxiliary results and definitions in Appendix A. Due to page constraints, the remaining of the paper discusses pseudo-deterministic learning only. Section 2 contains the proof of Theorem 1 and related results. A more general form of Theorem 2 is established in Section 3. For the proof of the other results and additional discussion, we refer to the full version of our paper.

[5] A form of approximate obfuscation is also investigated in [1], but their definition requires a much stronger correctness guarantee.

Algorithm A

Input: 1^n and oracle access to an unknown function $f \in \mathcal{C}_d(s(n))$.

1. Computes a multi-set $S_m \stackrel{\text{def}}{=} \{G_m(a) \mid a \in \{0,1\}^{\ell(m)}\}$ of m-bit strings (with multiplicities), where G_m is the pseudorandom generator with parameters as in the statement of the lemma.
2. For each $w \in S_m$, simulates D_n with oracle access to f and with its random input set to w. Let $h_w \stackrel{\text{def}}{=} D_n^f(w)$ be the hypothesis output by the learning circuit under f and w.
3. Outputs the description of a circuit $\widetilde{C}_f$ that on an input $x \in \{0,1\}^n$ computes the majority function over the multi-set $\{h_w(x) \mid w \in S_m\}$.

2 Pseudo-derandomization for randomized learning algorithms

In this section we consider the derandomization and pseudoderandomization of learning algorithms via pseudorandom generators and pseudodeterministic pseudorandom generators, respectively. For simplicity, we will mostly focus on self-learnable circuit classes, but our results can be extended to more general settings, as explained later in this section.

2.1 Derandomizing from a pseudorandom generator

We start with a technical lemma showing that standard pseudorandom generators can be used to derandomize learning algorithms.

▶ **Lemma 6** (PRG-based derandomization of learning algorithms). *Let $\mathfrak{C}$ be a circuit class closed under composition. Let $s, s' \colon \mathbb{N} \to \mathbb{N}$ be functions, where $s'(n) \geq n$. Further, let $\varepsilon, \delta > 0$ be real-valued parameters satisfying $\delta \leq \varepsilon \leq 1/100$ and possibly depending on n. Finally, assume that for each $n \geq 1$ the depth-d class $\mathcal{C}_d(s(n))$ can be (ε, δ)-learned by a (randomized) oracle $\mathcal{C}_{d'}(s'(n))$-circuit.*

There are constants $e = O(d \cdot d')$ and $k \geq 1$ for which the following holds. If there is a family of quick pseudorandom generators $G_m \colon \{0,1\}^{\ell(m)} \to \{0,1\}^m$ that ε-fool depth-e size-m $\mathfrak{C}$-circuits, for

$$m = O(s(n) \cdot s'(n) + s'(n)^k),$$

then $\mathcal{C}_d(s(n))$ can be deterministically learned to accuracy $\varepsilon' = 8\varepsilon$ in time at most $2^{O(\ell(m))} \cdot \mathsf{poly}(s'(n))$.[6]

Proof. Let $\{D_n\}_{n \geq 1}$ be the corresponding (uniform) sequence of learning circuits with fixed parameters ε and δ. We claim that the deterministic algorithm A learns every function $f \in \mathcal{C}_d(s(n))$ to accuracy ε' in time at most $2^{O(\ell(m))} \cdot \mathsf{poly}(s'(n))$.

Clearly, under our assumptions A is a deterministic algorithm that runs in time at most $2^{O(\ell(m))} \cdot \mathsf{poly}(s'(n))$. Suppose now that A fails to learn some function $f^\star \in \mathcal{C}_d[s(n)]$. In other words, the corresponding output hypothesis $\widetilde{C}_{f^\star}$ is not ε'-close to $f^\star$. We use this information to construct a *randomized* $\mathfrak{C}$-circuit B of size at most m and depth at most e that distinguishes the output of G_m from random with advantage at least ε. We then fix the randomness of B using a standard argument in order to obtain a deterministic distinguisher. This contradicts the pseudorandomness of G_m, completing the proof of the lemma.

[6] For unbounded-depth classes, the circuit depth parameters can be omitted from the statement.

Algorithm B

Input: $z \in \{0,1\}^m$ and $r \in \{0,1\}^n$.

1. Let $C_{f^\star}$ be a $\mathcal{C}_d(s(n))$-circuit that computes $f^\star$. B uses a prefix of z as the randomness of D_n, and simulates the oracle computation $D_n^{f^\star}(z)$ with $C_{f^\star}$ replacing its oracle gates.
2. Suppose h_z is the output hypothesis. B outputs 1 if and only if $h_z(r) = C_{f^\star}(r)$.

In the description of B presented next, z is a candidate string (either produced from the generator, or uniformly random), and r is a fixed string sampled according to $\boldsymbol{r} \sim \mathcal{U}_n$, a random variable representing the randomness of the distinguisher.

Since $C_{f^\star}$ has depth $\leq d$ and size $\leq s(n)$, and D_n is an oracle circuit of depth $\leq d'$ and size $\leq s'(n)$, Step 1 can be implemented by a $\mathfrak{C}$-circuit of depth at most $d \cdot d'$ and of size at most $s(n) \cdot s'(n)$. By definition, the output hypothesis of D_n is restricted to circuits in $\mathcal{C}_{d'}(s'(n))$, and h_z is an effective description of a $\mathfrak{C}$-circuit. Consequently, the evaluation $h_z(r)$ in Step 2 can be computed by a $\mathfrak{C}$-circuit of depth $O(d')$ and size $\mathsf{poly}(s'(n))$. It follows that Step 2 can be implemented by a $\mathfrak{C}$-circuit of depth no more than $O(d' + d)$ and of size no more than $O(s(n) + \mathsf{poly}(s'(n)))$. Overall, we get that B is a (randomized) $\mathfrak{C}$-circuit of depth at most e and of size at most m, where these parameters are as in the statement of the lemma.

We argue in what follows that

$$\left| \Pr_{\boldsymbol{x} \sim \mathcal{U}_m, \boldsymbol{r} \sim \mathcal{U}_n} [B(\boldsymbol{x}, \boldsymbol{r}) = 1] \; - \Pr_{\boldsymbol{y} \sim \mathcal{U}_{\ell(m)}, \boldsymbol{r} \sim \mathcal{U}_n} [B(G_m(\boldsymbol{y}), \boldsymbol{r}) = 1] \right| \; > \; \varepsilon. \tag{1}$$

Observe that this implies in particular that for some fixed choice of $r \in \{0,1\}^n$, $B_r \stackrel{\text{def}}{=} B(\cdot, r)$ is a *deterministic* $\mathfrak{C}$-circuit of no larger complexity that distinguishes $\mathcal{U}_m$ and $G_m(\mathcal{U}_{\ell(m)})$ with advantage at least ε, which completes the proof.

Consider the leftmost probability in Equation 1. Since D_n learns every $f \in \mathcal{C}_d(s(n))$ to accuracy ε and with confidence parameter δ and $C_{f^\star} \equiv f^\star$, with probability at least $1 - \delta$ over $\boldsymbol{x}$, Step 2 of circuit B computes a hypothesis $h_{\boldsymbol{x}}$ that is ε-close to $f^\star$. For each fixed $x \in \{0,1\}^m$ that produces an ε-close h_x, in Step 2 circuit B accepts the input pair (x, r) with probability at least $1 - \varepsilon$ over the choice of $r \sim \boldsymbol{r}$. Consequently, using that $\delta \leq \varepsilon$, the leftmost probability in Equation 1 is at least $(1 - \delta)(1 - \varepsilon) \geq 1 - 2\varepsilon$.

It remains to upper bound the rightmost probability. Because A fails to learn $f^\star$ to accuracy ε', there is a set $T \subseteq \{0,1\}^n$ of measure at least ε' such that on every $x \in T$, $\widetilde{C}_{f^\star}(x) \neq f^\star(x)$. Consequently, for $x \in T$ at least half of the values $h_w(x)$ generated in Step 3 of A's description do not agree with $f^\star(x)$. It follows that over the choice of $\boldsymbol{y}$ and $\boldsymbol{r}$, $B(G_m(\boldsymbol{y}), \boldsymbol{r})$ rejects with probability at least $\varepsilon'/2 = 4\varepsilon$. Consequently, the rightmost probability $\leq 1 - 4\varepsilon$.

It follows from these estimates that the distinguishing probability in Equation 1 is strictly larger than ε, from which the result follows. ◀

It is important in the preceding argument for the distribution employed in the derandomization to be pseudorandom against *non-uniform* $\mathfrak{C}$-circuits.[7] First, this allows us to disregard the complexity of uniformly generating D_n in the proof that B is an appropriate distinguisher. Most importantly, we have no control over the "bad" function $f^\star$ where the derandomization

[7] Distributions that are pseudorandom against uniform algorithms were crucially employed in the pseudodeterministic construction of primes from [29].

might fail, and consequently $C_{f^\star}$ appears as a non-uniform advice in the proof of Lemma 6. Finally, r is also fixed non-uniformly when derandomizing the distinguisher.

2.2 Pseudoderandomization of learning algorithms

Similarly to Lemma 6, we now show that self-learning classes admit pseudodeterministic learners under the existence of suitable pseudodeterministic pseudorandom generators.

▶ **Lemma 7** (Pseudoderandomization via pseudodeterministic PRGs). *Let $\mathfrak{C}$ be a circuit class closed under composition. Let $s, s' \colon \mathbb{N} \to \mathbb{N}$ be functions, where $s'(n) \geq n$. Further, let $\varepsilon, \delta, \mu > 0$ be real-valued parameters satisfying $\delta \leq \varepsilon \leq 1/100$ and possibly depending on n. Finally, assume that for each $n \geq 1$ the depth-d class $\mathcal{C}_d(s(n))$ can be (ε, δ)-learned by a (randomized) oracle $\mathcal{C}_{d'}(s'(n))$-circuit.*

There are constants $e = O(d \cdot d')$ and $k \geq 1$ for which the following holds. If there is a family of quick μ-pseudodeterministic pseudorandom generators $G_m \colon \{0,1\}^{t(m)} \times \{0,1\}^{\ell(m)} \to \{0,1\}^m$ that ε-fool depth-e size-m $\mathfrak{C}$-circuits, for

$$m = O(s(n) \cdot s'(n) + s'(n)^k),$$

then $\mathcal{C}_d(s(n))$ can be $(8\varepsilon, \mu, \mu)$-pseudodeterministically learned in randomized time at most $2^{O(\ell(m))} \cdot \mathsf{poly}(s'(n))$.

Proof. We proceed as in the proof of Lemma 6, except that the corresponding derandomized algorithm A is replaced here by a pseudoderandomized algorithm A'. This procedure uses its random input $\boldsymbol{y} \in \{0,1\}^{t(m)}$ to define a candidate (deterministic) pseudorandom generator $G_m^{\boldsymbol{y}} \stackrel{\text{def}}{=} G_m(\boldsymbol{y}, \cdot) \colon \{0,1\}^{\ell(m)} \to \{0,1\}^m$. By assumption, it succeeds with probability at least $1 - \mu$, and whenever this happens, A' outputs a hypothesis $h_{\boldsymbol{y}}$ that is ε'-close to f, the unknown function, where $\varepsilon' = 8\varepsilon$ is as in Lemma 6. Consequently, A' is a (ε', μ)-learner for the class. Furthermore, with probability at least $1 - \mu$, A' constructs the same pseudorandom generator. Since the rest of its computation is deterministic, the corresponding learner will make a fixed set Q_f of queries, and generate a fixed output hypothesis h_f. This shows that A' is μ-pseudodeterministic. As the running time of A and A' are the same up to low order terms, it follows that $\mathcal{C}_d(s(n))$ can be $(8\varepsilon, \mu, \mu)$-pseudodeterministically learned in randomized time at most $2^{O(\ell(m))} \cdot \mathsf{poly}(s'(n))$. ◀

▶ Remark. As we alluded to before, Lemmas 6 and 7 hold in more generality provided that we have sufficiently strong pseudorandom generators. In particular, it is sufficient to have a generator that fools a circuit class closed under composition that is expressive enough to simulate circuits in the concept class, the learning circuit, and its hypothesis class. Consequently, existing learning algorithms can be derandomized under hardness assumptions.

▶ **Theorem 8** (Conditional learning derandomization). *Let $\mathfrak{C} \subseteq \mathsf{P/poly}$ be an arbitrary circuit class, and suppose $\mathfrak{C}(s(n))$ can be learned to any constant accuracy by a randomized algorithm running in time $t(n) \geq n$. If $\mathsf{E} = \mathsf{DTIME}[2^{O(n)}]$ requires circuits of size $2^{\Omega(n)}$ on all large input lengths, then there exists a constant $c \geq 1$ such that $\mathfrak{C}(s)$ can be deterministically learned to any constant accuracy in time at most $O(t(n)^c)$.*

Proof. Observe that a learning algorithm running in time $t(n)$ can be implemented by oracle circuits of size at most $\mathsf{poly}(t(n))$. The result is then a direct consequence of Lemma 6 and the hardness vs. randomness paradigm (Theorem 17). ◀

As a concrete example, Theorem 8 and Jackson's polynomial time learning algorithm for DNFs [19] immediately imply the following result.

▶ **Corollary 9.** *If* $\mathsf{E} = \mathsf{DTIME}[2^{O(n)}]$ *requires circuits of size* $2^{\Omega(n)}$ *on all large input lengths, then polynomial size* DNF*s can be learned to constant accuracy in deterministic polynomial time.*

The same approach provides pseudoderandomization via Lemma 7 using that a hard truth-table can be *pseudodeterministically* constructed from a weaker lower bound assumption.

▶ **Theorem 10** (Conditional learning pseudoderandomization)**.** *Let* $\mathfrak{C} \subseteq \mathsf{P/poly}$ *be an arbitrary circuit class, and suppose* $\mathfrak{C}(s(n))$ *can be learned to any constant accuracy by a randomized algorithm running in time* $t(n) \geq n$*. If* $\mathsf{BPE} = \mathsf{BPTIME}[2^n]$ *requires circuits of size* $2^{\Omega(n)}$ *on all large input lengths, then there exists a constant* $c \geq 1$ *such that* $\mathfrak{C}(s)$ *can be pseudodeterministically learned to any constant accuracy in time at most* $O(t(n)^c)$*.*

Proof. Note that, under this lower bound assumption, there exists a randomized algorithm that on input 1^ℓ, runs in time at most $2^{O(\ell)}$ and outputs with high probability the description of a fixed function $f_\ell \colon \{0,1\}^\ell \to \{0,1\}$ that requires circuits of size $2^{\Omega(\ell)}$. In other words, exponentially hard boolean functions can be pseudo-deterministically constructed in time polynomial in the size of their truth tables. The result now follows from the learning assumption, Theorem 17, and Lemma 7. ◀

For instance, thanks to the quasi-polynomial time randomized learning algorithm for $\mathcal{AC}^0[p]$ from [4], we get the following conditional result.

▶ **Corollary 11.** *If there is* $\gamma > 0$ *and a language in* $\mathsf{BPE} = \mathsf{BPTIME}[2^n]$ *that requires circuits of size* $\geq 2^{n^\gamma}$ *on all large input lengths, then* $\mathcal{AC}^0[p]$ *circuits can be learned to any constant accuracy in pseudodeterministic quasi-polynomial time.*

Proof. Simply observe that this lower bound is enough to get quasi-polynomial time (pseudo-deterministic) derandomizations using the hardness versus randomness paradigm (Theorem 17). The result follows as in the proof of Theorem 10 using the learning algorithm from [4]. ◀

3 Pseudodeterministic learners from randomized learners

Recall that $\mathcal{AC}^0$ circuits can be deterministically learned in quasi-polynomial time [33], and that $\mathcal{AC}^0[p]$ circuits are known to be learnable in randomized quasi-polynomial time [4]. In this section, we prove a general result showing that, for strong enough self-learnable circuit classes, any randomized learner running in quasi-polynomial time admits a non-trivial pseudoderandomization.

As opposed to the results discussed in Section 2, the next theorem is *unconditional* and does not assume the existence of pseudorandom generators. Theorem 2 is a particular case of this result.

▶ **Theorem 12** (Pseudodeterministic learners from randomized self-learners)**.** *Let* $\mathfrak{C}$ *be a circuit class that contains* $\mathcal{TC}^0$ *and is closed under compositions. Suppose that for every* $\delta, \varepsilon > 0$*,* $\mathfrak{C}(\mathsf{poly})$ *can be* (ε, δ)*-learned by (uniform)* $\mathfrak{C}$*-circuits of quasi-polynomial size. Then, for every* $\gamma > 0$ *and* $c \geq 1$*,* $\mathfrak{C}(\mathsf{poly})$ *can be* μ*-pseudodeterministically learned to accuracy* $\leq n^{-c}$ *on infinitely many input lengths by an algorithm running in time* $O(2^{n^\gamma})$*, where* $\mu = 2^{-n}$*.*

Proof. First, the assumption implies by a padding argument that for every $\varepsilon > 0$ and $k \geq 1$, there exists $k' \geq 1$ such that $\mathcal{C}(\exp((\log n)^k))$ can be learned to accuracy ε by a uniform

family $\mathfrak{D}^{(k)} = \{D_n^{(k)}\}_{n\geq 1}$ of $\mathfrak{C}$-circuits of size at most $\exp((\log n)^{k'})$.[8] We recall the following result from [23], which for convenience we state here as follows.

▶ **Lemma 13.** *There is a* PSPACE*-complete language L computable in linear space and a constant $b \geq 1$ such that the following holds. If $\mathfrak{C}(s(n))$ is learnable to error and accuracy $\leq n^{-b}$ in time at most $t(n) \geq n$, then either*

(i) *$L \notin \mathfrak{C}(s(n))$; or*

(ii) *$L \in \mathsf{BPTIME}[\mathsf{poly}(t(n))]$.*

By amplifying the success probability, we can assume that each family $\mathfrak{D}^{(k)}$ learns with confidence parameter $\delta \leq n^{-b}$, and by the result of [2], we can assume without loss of generality that the accuracy parameter is also $\leq n^{-b}$. This implies via Lemma 13 that either there exists no constant $a \geq 1$ such that $L \in \mathfrak{C}(\exp((\log n)^a))$, or for some $a' \geq 1$, we have $L \in \mathsf{BPTIME}[\exp((\log n)^{a'})]$. In the former case, since L is computable in linear space, we get that $\mathsf{BPE} \not\subseteq \mathfrak{C}[\exp((\log n)^{O(1)})]$. On the other hand, in the latter scenario, as $\mathsf{DSPACE}[s'(n)]$ can diagonalize against circuits of size $s'(n)^{\Omega(1)}$ (cf. [28, Corollary 39]), this fact together with a standard padding argument implies that $\mathsf{BPE} \not\subseteq \mathfrak{C}[\exp((\log n)^{O(1)})]$.

Let $f \in \mathsf{BPE}$ be a function that cannot be computed by quasi-polynomial size circuits from $\mathfrak{C}$ on infinitely many input lengths. We claim that the following result holds.

▶ **Lemma 14.** *Under our assumptions, there exists a function $f' \in \mathsf{BPTIME}[2^{O(n)}]$ such that for every constant $\beta \geq 1$, on infinitely many input lengths n, any $\mathfrak{C}$-circuit of size $\leq \exp((\log n)^{\beta})$ can compute f'_n with advantage at most $\exp((\log n)^{-\beta})$.*

Indeed, since $\mathcal{TC}^0 \subseteq \mathfrak{C}$, efficient worst-case to average-case reductions can be used to amplify the hardness of f (cf. [11, 14, 8]). In a bit more detail, a reduction of this form is well-known to hold for functions $f \in \mathsf{E} = \mathsf{DTIME}[2^{O(n)}]$. In order to amplify a function f in BPE, it is enough to observe that the entire truth-table of f can be computed in randomized time $2^{O(n)}$, except with negligible probability. Since the worst-case to average-case reduction acts on truth-tables, it defines with high probability a fixed function f' obtained from f.

Let $f' = \{f'_n\}_{n\geq 1}$ be given by Lemma 14, and E be a randomized algorithm running in time $2^{O(n)}$ that prints the truth-table of f'_n with probability at least $1 - 2^{-n}$. We use E together with the Nisan-Wigderson generator [26] to pseudodeterministically compute a generator against $\mathfrak{C}$. (While their result is stated with respect to general boolean circuits, it is well-known and easy to check that their construction works for any circuit class containing $\mathcal{TC}^0$.)

▶ **Theorem 15** (Corollary of Theorem 1 from [26])**.** *Let $m \leq t(m) \leq 2^m$, and suppose there is $h \in \mathsf{DTIME}[2^{O(m)}]$ such that, on infinitely many input lengths, every $\mathfrak{C}$-circuit D_m of size $\leq t(m)$ satisfies $\Pr_{\boldsymbol{x}}[D_m(\boldsymbol{x}) \neq h_m(\boldsymbol{x})] \geq 1/m$. Then there exists a constant $\lambda > 0$ and a quick pseudorandom generator $G\colon \{0,1\}^m \to \{0,1\}^{t(m^\lambda)}$ that $t(m^\lambda)$-fools $\mathcal{C}(t(m^\lambda))$-circuits on infinitely many input lengths.*

Using Theorem 15, it is possible to prove the following result.

▶ **Lemma 16.** *For every constants $c \geq 1$, $k \geq 1$, and $\gamma > 0$, there exists a function $G\colon \{0,1\}^* \times \{0,1\}^{\ell(n)} \to \{0,1\}^n$ that is a quick μ-pseudodeterministic generator that η-fools $\mathfrak{C}$-circuits of size $\leq \exp((\log n)^k)$ on infinitely many input lengths, where $\mu = 2^{-n}$, $\eta = n^{-c}$, and $\ell(n) = n^{\gamma}$.*

[8] See for instance the proof of Lemma 7 in [28].

Lemma 16 is established by a standard application of the Nisan-Wigerson generator to the family f', adapted to the pseudo-deterministic setting in the natural way.

Finally, using that $\mathfrak{C}$ is closed under composition, the existence of such generators immediately imply the statement of the theorem via an application of Lemma 7. ◀

Ideally, we would like to obtain a pseudodeterministic learner of comparable running time. However, this does not seem to be possible with these techniques. Consider for instance the more extreme case of designing a sub-exponential time pseudodeterministic learner from a sub-exponential time randomized learner. The main difficulty is that the lower bounds obtained from such a learner are not strong enough to derandomize an algorithm that runs in sub-exponential time.

Our techniques also require a strong assumption on the circuit class, namely, that it is closed under composition and able to compute threshold functions. Since there is evidence that circuit classes containing $\mathcal{TC}^0$ cannot be learned [25], it would be extremely interesting to obtain an analogue of Theorem 12 under weaker assumptions. In particular, one might be able to apply such a result to pseudoderandomize existing algorithms, such as [4].

Two remarks on the self-learnability of weak classes. These results further motivate the study of self-learning circuit classes, a direction that some might find of independent philosophical interest. In other words,

When is a circuit class $\mathfrak{C}$ learnable by algorithms that are no more powerful than $\mathfrak{C}$-circuits?

For very weak classes, this is probably impossible, given the very weak resources available to the learning algorithm, and the fact that a self-learner is in particular a proper learner. However, when $\mathfrak{C}$ becomes stronger, as in the extreme case where $\mathfrak{C} = \mathsf{P/poly}$, if learning algorithms exist then they are automatically proper learners.

It is possible to show that $\mathsf{MAJ} \circ \mathcal{AC}^0$ circuits are self-learnable by a uniform family of sub-exponential size circuits. This follows for instance from the results of [12], since the learning algorithm is based on the estimation of fourier coefficients of bounded size, and the corresponding parity computations can be simulated by randomized oracle $\mathcal{AC}^0$ circuits that output a sub-exponential size hypothesis in $\mathsf{MAJ} \circ \mathcal{AC}^0$.[9] Therefore, self-learnability is a phenomenon that is present even in constant-depth classes.

On the other hand, we do not know if $\mathcal{AC}^0$ is self-learnable by quasi-polynomial size $\mathcal{AC}^0$ circuits.[10] A natural approach here is to try to implement the LMN algorithm [24] using $\mathcal{AC}^0$ circuits, perhaps by replacing a threshold gate for an approximate majority, which is known to be computable in this class (see e.g. [35]). However, as we briefly explain next, this and other similar approaches cannot work. A self-learning algorithm of quasi-polynomial complexity for the class $\mathcal{AC}^0$ is required to output a hypothesis that is itself a quasi-polynomial size $\mathcal{AC}^0$ circuit. However, the approach in [24] is also able to learn functions that cannot be approximated by such circuits. This is an immediate consequence of [31, Theorem 3] using the connection between the influence of a boolean function and fourier concentration (cf. [27]).

[9] For the interested reader, we stress that during this implementation the empirical estimate of each fourier coefficient is not computed by the circuit, since $\mathcal{AC}^0$ circuits cannot count. The bits obtained from products of the form $\chi_S(x) \cdot f(x)$ are hard-coded directly into the final hypothesis, which can make use of a single threshold gate to compute the sign function.

[10] Note that, if this were the case, our techniques from Section 2 would provide an alternative, conceptually simpler proof of a result of [32] showing that such circuits can be learned in deterministic quasi-polynomial time.

References

1. Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. *Journal of the ACM*, 59(2):6:1–6:48, 2012.
2. Dan Boneh and Richard J. Lipton. Amplification of weak learning under the uniform distribution. In *Conference on Computational Learning Theory (COLT)*, pages 347–351, 1993.
3. Zvika Brakerski, Christina Brzuska, and Nils Fleischhacker. On statistically secure obfuscation with approximate correctness. In *International Cryptology Conference (CRYPTO)*, pages 551–578, 2016.
4. Marco L. Carmosino, Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. Learning algorithms from natural proofs. In *Conference on Computational Complexity (CCC)*, pages 10:1–10:24, 2016.
5. Bill Fefferman, Ronen Shaltiel, Christopher Umans, and Emanuele Viola. On beating the hybrid argument. *Theory of Computing*, 9:809–843, 2013.
6. Lance Fortnow and Adam R. Klivans. Efficient learning algorithms yield circuit lower bounds. *Journal of Computer and System Sciences*, 75(1):27–36, 2009. doi:10.1016/j.jcss.2008.07.006.
7. Eran Gat and Shafi Goldwasser. Probabilistic search algorithms with unique answers and their cryptographic applications. *Electronic Colloquium on Computational Complexity (ECCC)*, 18:136, 2011.
8. Oded Goldreich, Noam Nisan, and Avi Wigderson. On Yao's XOR lemma. In *Studies in Complexity and Cryptography*, pages 273–301. Springer, 2011. doi:10.1007/978-3-642-22670-0_23.
9. Shafi Goldwasser and Ofer Grossman. Bipartite perfect matching in pseudo-deterministic NC. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 87:1–87:13, 2017.
10. Shafi Goldwasser, Ofer Grossman, and Dhiraj Holden. Pseudo-deterministic proofs. *Electronic Colloquium on Computational Complexity (ECCC)*, 24:105, 2017.
11. Shafi Goldwasser, Dan Gutfreund, Alexander Healy, Tali Kaufman, and Guy N. Rothblum. Verifying and decoding in constant depth. In *Symposium on Theory of Computing (STOC)*, pages 440–449, 2007.
12. Parikshit Gopalan and Rocco A. Servedio. Learning and lower bounds for AC^0 with threshold gates. In *International Workshop on Approximation, Randomization, and Combinatorial Optimization (RANDOM-APPROX)*, pages 588–601, 2010.
13. Ofer Grossman. Finding primitive roots pseudo-deterministically. *Electronic Colloquium on Computational Complexity (ECCC)*, 22:207, 2015.
14. Dan Gutfreund and Guy N. Rothblum. The complexity of local list decoding. In *International Workshop on Approximation, Randomization and Combinatorial Optimization (RANDOM-APPROX)*, pages 455–468, 2008.
15. Ryan C. Harkins and John M. Hitchcock. Exact learning algorithms, betting games, and circuit lower bounds. *Transactions on Computation Theory (TOCT)*, 5(4):18:1–18:11, 2013. doi:10.1145/2539126.2539130.
16. Dhiraj Holden. A note on unconditional subexponential-time pseudo-deterministic algorithms for BPP search problems. *arXiv*, 2017. arXiv:1707.05808.
17. Russell Impagliazzo, Valentine Kabanets, and Avi Wigderson. In search of an easy witness: exponential time vs. probabilistic polynomial time. *Journal of Computer and System Sciences*, 65(4):672–694, 2002. doi:10.1016/S0022-0000(02)00024-7.

18 Russell Impagliazzo and Avi Wigderson. P = BPP if E requires exponential circuits: Derandomizing the XOR lemma. In *Symposium on the Theory of Computing (STOC)*, pages 220–229, 1997. doi:10.1145/258533.258590.

19 Jeffrey C. Jackson. An efficient membership-query algorithm for learning DNF with respect to the uniform distribution. *Journal of Computer and System Sciences*, 55(3):414–440, 1997.

20 Mark Jerrum, Alistair Sinclair, and Eric Vigoda. A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries. *Journal of the ACM*, 51(4):671–697, 2004.

21 Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004. doi:10.1007/s00037-004-0182-6.

22 Michael Kearns and Umesh Vazirani. *An Introduction to Computational Learning Theory*. MIT Press, 1994.

23 Adam Klivans, Pravesh Kothari, and Igor C. Oliveira. Constructing hard functions using learning algorithms. In *Conference on Computational Complexity (CCC)*, pages 86–97, 2013.

24 Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, fourier transform, and learnability. *Journal of the ACM*, 40(3):607–620, 1993.

25 Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. *Journal of the ACM*, 51(2):231–262, 2004.

26 Noam Nisan and Avi Wigderson. Hardness vs randomness. *Journal of Computer and System Sciences*, 49(2):149–167, 1994.

27 Ryan O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014.

28 Igor C. Oliveira and Rahul Santhanam. Conspiracies between learning algorithms, circuit lower bounds, and pseudorandomness. In *Computational Complexity Conference (CCC)*, pages 18:1–18:49, 2017.

29 Igor C. Oliveira and Rahul Santhanam. Pseudodeterministic constructions in subexponential time. In *Symposium on Theory of Computing (STOC)*, pages 665–677, 2017.

30 Alexander A. Razborov and Steven Rudich. Natural proofs. *Journal of Computer and System Sciences*, 55(1):24–35, 1997.

31 Benjamin Rossman, Rocco A. Servedio, and Li-Yang Tan. An average-case depth hierarchy theorem for boolean circuits. *CoRR*, abs/1504.03398, 2015.

32 Meera Sitharam. Pseudorandom generators and learning algorithms for AC^0. *Computational Complexity*, 5(3/4):248–266, 1995.

33 Meera Sitharam and Timothy Straney. Derandomized learning of boolean functions. In *International Conference on Algorithmic Learning Theory (ALT)*, pages 100–115, 1997.

34 Christopher Umans. Pseudo-random generators for all hardnesses. *Journal of Computer and System Sciences*, 67(2):419–440, 2003.

35 Emanuele Viola. Randomness buys depth for approximate counting. *Computational Complexity*, 23(3):479–508, 2014.

36 Ilya Volkovich. On learning, lower bounds and (un)keeping promises. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 1027–1038, 2014. doi:10.1007/978-3-662-43948-7_85.

37 Ryan Williams. Improving exhaustive search implies superpolynomial lower bounds. *SIAM Journal on Computing*, 42(3):1218–1244, 2013. doi:10.1137/10080703X.

A Learning, Approximation, and Auxiliary Results

Let $\mathcal{F}_n$ be the set of all boolean functions $f\colon \{0,1\}^n \to \{0,1\}$ on n input variables, and $\mathfrak{F} = \bigcup_{n\geq 1} \mathcal{F}_n$ be the set of all boolean functions. We use boldface letters such as $\boldsymbol{w}$ and $\boldsymbol{x}$ to denote random variables. We say that boolean functions f and g from $\mathcal{F}_n$ are ε-close if $\Pr_{\boldsymbol{x}\sim\mathcal{U}_n}[f(\boldsymbol{x}) \neq g(\boldsymbol{x})] \leq \varepsilon$, where $\mathcal{U}_n$ denotes the uniform distribution over $\{0,1\}^n$. We often view a string in $\{0,1\}^*$ that represents a boolean circuit D as if it were the actual circuit D, or the function that it computes.

A.1 Randomness, pseudorandomness and pseudodeterminism

We will require the notion of polynomial-time samplability of a sequence of distributions. Let $\mathfrak{D} = \{\mathcal{D}_n\}$ be a sequence of distributions, where each $\mathcal{D}_n$ is supported on $\{0,1\}^n$. We say that $\mathfrak{D}$ is polynomial-time samplable if there is a probabilistic polynomial-time algorithm B such that for each $n \in \mathbb{N}$ and each $y \in \{0,1\}^*$, $\Pr_{\boldsymbol{B}}[\boldsymbol{B}(1^n) = y] = \Pr[y \in \mathcal{D}_n]$.

We also require notions of pseudorandomness, introduced next.

Pseudorandom generators and hitting set generators. Let $\mathcal{D}_m$ be a probability distribution supported over $\{0,1\}^m$. We say that $\mathcal{D}_m$ is (η, s)-*pseudorandom* for a circuit class $\mathcal{C} \subseteq \mathcal{F}_m$ if for each size-s circuit $g \in \mathcal{C}(s)$,

$$\left| \Pr_{\boldsymbol{x}\sim\mathcal{U}_m}[g(\boldsymbol{x}) = 1] - \Pr_{\boldsymbol{y}\sim\mathcal{D}_m}[g(\boldsymbol{y}) = 1] \right| < \eta.$$

In other words, the circuit g is η-*fooled* by $\mathcal{D}_m$. This definition extends to an ensemble $\mathfrak{D} = \{\mathcal{D}_m\}_{m\geq 1}$ of distributions, by requiring the condition above to hold for every $m \geq 1$. Moreover, we say that a function $G_m\colon \{0,1\}^{\ell(m)} \to \{0,1\}^m$ is an η-*pseudorandom generator* for a class $\mathcal{C}$ if the induced distribution $G_m(\mathcal{U}_{\ell(m)})$ is (η, m)-pseudorandom for $\mathcal{C}$. Equivalently, the induced distribution η-fools every size-m $\mathcal{C}$-circuit over m-input variables. The function $\ell(m)$ computes the *seed length* of the generator G_m.

We also consider the weaker notion of hitting sets. We say that a set $\mathcal{H}_m \subseteq \{0,1\}^m$ is an (η, s)-*hitting set* for $\mathcal{C}$ if for each size-s circuit $g \in \mathcal{C}(s)$ such that $\Pr_{\boldsymbol{x}\sim\mathcal{U}_m}[g(\boldsymbol{x}) = 1] \geq \eta$, we have $g^{-1}(1) \cap \mathcal{H}_n \neq \emptyset$. This definition extends to ensembles of sets in the natural way. Similarly, a function $H_m\colon \{0,1\}^{\ell(m)} \to \{0,1\}^m$ is an η-*hitting set* for $\mathcal{C}$ if the induced set $H_m(\{0,1\}^{\ell(m)}) \subseteq \{0,1\}^m$ is an (η, m)-hitting set for $\mathcal{C}$. (Note that the support of a pseudorandom distribution is a hitting set with the same parameter η.)

We say that a pseudorandom generator or a hitting set generator is *quick* if it can be computed in time $2^{O(\ell(m))}$, where $\ell(m)$ is the corresponding seed length.

The following result will be useful.

▶ **Theorem 17** ([34]). *Given a function $f\colon \{0,1\}^{\log \ell} \to \{0,1\}$ of circuit complexity at least s, it is possible to construct a pseudorandom generator $G\colon \{0,1\}^{O(\log \ell)} \to \{0,1\}^m$ that $(1/m)$-fools size m circuits, where $m = s^{\Omega(1)}$. Moreover, G can be computed in time $\ell^{O(1)}$ given the description of the truth table of f.*

Pseudodeterministic pseudorandommness. We will make use of pseudorandom distributions $\mathcal{D}_m$ and hitting sets $\mathcal{H}_m$ that are constructed *pseudodeterministically*. For our purposes, we define the relevant concepts as follows. Let $G_m\colon \{0,1\}^{t(m)} \times \{0,1\}^{\ell(m)} \to \{0,1\}^m$. We say that G_m is a μ-pseudodeterministic (η, m)-pseudorandom generator for $\mathcal{C}$ if there is an (η, m)-pseudorandom generator $G^\star_m\colon \{0,1\}^{\ell(m)} \to \{0,1\}^m$ for $\mathcal{C}$ such that

$$\Pr_{\boldsymbol{a}\sim\mathcal{U}_t}[G(\boldsymbol{a}, \cdot) \equiv G^\star_m(\cdot)] \;\geq\; 1 - \mu,$$

where the "≡" symbol represents identity among functions. A μ-pseudodeterministic hitting set generator $H_m\colon \{0,1\}^{t(m)} \times \{0,1\}^{\ell(m)} \to \{0,1\}^m$ is defined analogously. Analogously, we say that a pseudodeterministic pseudorandom or hitting set generator is *quick* if it can be computed in time $2^{O(\ell(m))}$, where $\ell(m)$ is the seed length.

Note that the pseudodeterministic parameter μ of a quick pseudorandom or hitting set generator can be boosted by standard techniques. Indeed, since quick generators can tolerate a running time overhead of $2^{O(\ell(n))}$, one can always design a new generator that uses a larger random string, samples independent copies of the initial pseudodeterministic generator, and behaves as the most common generator among the induced generators provided by these samples.

A.2 Learning

Learning algorithms. We consider randomized learning algorithms under the uniform distribution that can make membership queries to the unknown function. We formalize such algorithms next.

Fix a class of functions $\mathfrak{C} \subseteq \mathfrak{F}$, often referred to as the *concept class*. For convenience, we write $\mathfrak{C} = \{\mathcal{C}_n\}_{n\geq 1}$, where $\mathcal{C}_n \subseteq \mathcal{F}_n$. A randomized algorithm A (ε,δ)-*learns* a class $\mathfrak{C}$ if for every $n \geq 1$ and for each $f \in \mathcal{C}_n$, when given oracle access to f and access to inputs 1^n, $\varepsilon > 0$ (accuracy), and $\delta > 0$ (confidence), A outputs the description of a boolean circuit D such that

$$\Pr_{\boldsymbol{w}}[\boldsymbol{D} = A^f(1^n,\varepsilon,\delta,\boldsymbol{w}) \text{ is } \varepsilon\text{-close to } f\,] \geq 1-\delta.$$

Here $\boldsymbol{w} \in \{0,1\}^*$ is a uniformly random boolean string representing the randomness of A, and $\boldsymbol{D} = A^f(1^n, w, \varepsilon, \delta)$ is a random variable denoting the (representation of the) circuit output by A over these inputs and with oracle access to f. For convenience, we might omit some input parameters when discussing the computation of A.[11]

While many of our results hold in a more general setting, for simplicity we will focus on the learnability of classes of boolean circuits. Therefore, $\mathcal{C}_n$ will always denote a class of the form $\mathcal{C}(s(n))$, where $\mathcal{C} \in \{\mathcal{AC}^0, \mathcal{AC}^0[p], \mathcal{TC}^0, \text{etc.}\}$, and $s(n)$ is an upper bound on circuit size complexity. When there is no risk of confusion, we might write $\mathcal{C}_d$ to restrict the class to circuits of depth at most d. The worst-case running time of the learning algorithm A over the choice of $f \in \mathcal{C}(s(n))$ and of its internal random string w is measured by the function $t_A(n, s, 1/\varepsilon, 1/\delta)$.

Pseudodeterministic learning. A randomized algorithm A $(\varepsilon,\delta,\gamma)$-*pseudodeterministically learns* a class $\mathfrak{C} = \{\mathcal{C}_n\}$ if A (ε,δ)-learns this class, and moreover for every $n \geq 1$ and $f \in \mathcal{C}_n$ there is a fixed set of queries $Q_f \subseteq \{0,1\}^n$ and a fixed string D_f representing a boolean circuit such that

$$\Pr_{\boldsymbol{w}}[A^f(1^n,\boldsymbol{w}) \text{ queries } f \text{ exactly over } Q_f \text{ and } A^f(1^n,\boldsymbol{w}) = D_f] \geq 1-\gamma.$$

In other words, with high probability the learner makes the same set of queries and outputs the same boolean circuit (representation) as its hypothesis. We say in this case that A is a γ-*pseudodeterministic* learning algorithm.

[11] It is well-known that the confidence parameter δ can be made arbitrarily small (cf.[22]). It is also known how to boost the accuracy parameter ε if the concept class satisfies a certain closure property (see e.g. [2]).

We would like to stress that it makes sense to consider variants of this notion where only the set of queries is pseudodeterministic (*query-pseudodet. learner*), or where only the output hypothesis is pseudodeterministic (*hypothesis-pseudodet. learner*). For instance, if A is a pseudodeterministic learner, running several independent copies of A and outputting the most common hypothesis will boost the initial hypothesis-pseudodeterminism parameter, but the resulting learner will be no longer query-pseudodeterministic.

The circuit complexity of learning algorithms. It is crucial in our investigations to consider a notion of complexity for learning algorithms that is more refined than running time. We measure instead the *circuit complexity* of learning algorithms. In other words, we specify a learning algorithm by a sequence $\{D_n\}_{n\geq 1}$ of *multi-output oracle* circuits D_n that have access to $\boldsymbol{w}$, ε, and δ, and whose *oracle queries* are answered according to the unknown function $f \in \mathcal{C}_n$. (In particular, the main input string of the oracle circuit is the random string, and in many cases we will fix ε and δ in advance.) The output bits of D_n encode a circuit describing the output hypothesis.[12]

In the case of learning circuits that are less powerful than general circuits (such as $\mathcal{AC}^0$, $\mathcal{TC}^0$, etc.), we further restrict the output hypothesis of the learner. We say that a class $\mathfrak{C}$ is learnable by $\mathfrak{D}$-circuits if the sequence $\{D_n\}$ consists of circuits from $\mathfrak{D}$, and moreover the output string is an *effective encoding* of a $\mathfrak{D}$-circuit. The meaning of effective description is that it should be possible for $\mathfrak{D}$-circuits to interpret the output string as the description of a $\mathfrak{D}$-circuit, and to efficiently evaluate computations given this description. We will not be explicit about such encodings, and simply note that they exist for the typical circuit classes investigated in our work.[13]

For definiteness, we briefly discuss a notion of *uniformity* for such sequence of learning circuits. In *learning upper bounds*, we assume that the sequence can be generated from 1^n by a deterministic algorithm that runs in time polynomial in the size of the circuits. We will not discuss *circuit lower bounds for learning* in this paper, but in such a context it is also natural to consider *non-uniform* sequences of learning circuits (see e.g. [28, Section 4]).

We say that a circuit class $\mathfrak{C}$ is *self-learnable* if it can be learned by a sequence of $\mathfrak{C}$-circuits, typically of quasi-polynomial size. This is an informal working definition, since for instance we do not specify the dependence on the parameters ε and δ. We will leave such details to the formal statement of our results, where we will often assume that these learning parameters are sufficiently small constants, and allow the class $\mathcal{C}_d(n^k)$ to be learned by $\mathcal{C}_{d'}(n^{(\log n)^{k'}})$-circuits (multi-output and with oracle gates).

We assume that functions related to algorithmic parameters such as time bounds, circuit size, learning accuracy, etc. are sufficiently constructive, in the sense that they do not affect the asymptotic complexity of our reductions whenever an algorithm needs to compute one of these functions.

A.3 Approximation

Approximation schemes. We define notions of approximation for computing integer-valued functions. An *integer-valued function* is a function from strings to non-negative integers,

[12] In the case of *deterministic* learning circuits, we remark that each D_n has access to the constant input bits 0 and 1, and one can think of its "input string" as the first batch of answers provided by the oracle queries.

[13] For bounded-depth circuit classes, we tolerate a constant-factor depth blow-up during the evaluation if this is necessary from the choice of encoding.

i.e., from $\{0,1\}^*$ to $\mathbb{N}$. We say that an integer-valued function f has a *polynomial-time randomized approximation scheme* (PRAS) if for each rational number $\epsilon > 0$ there is a probabilistic polynomial-time machine M, which given any string x as input, outputs an integer $M(x)$ (which might depend on the random choices of M) such that with probability $1 - 2^{-\Omega(|x|)}$ over the random choices of M, we have that $(1-\epsilon)f(x) \leq M(x) \leq (1+\epsilon)f(x)$. We say that f has an *fully polynomial-time randomized approximation scheme* (FPRAS) if there is a probabilistic machine M, which given a string x and a rational number ϵ (in some prespecified format) as input, runs in time $\text{poly}(|x|, 1/\epsilon)$ and outputs a number $M(x)$ such that with probability $1 - 2^{-\Omega(|x|)}$ over the random choices of M, we have that $(1-\epsilon)f(x) \leq M(x) \leq (1+\epsilon)f(x)$.

An example of an integer-valued function is the permanent of a $(0,1)$-matrix, when the matrix is represented as a bitstring. By the celebrated result of Jerrum, Sinclair and Vigoda [20], this integer-valued function has an FPRAS.

We will be interested in converting randomized approximation schemes to *pseudo-deterministic* ones, where with high probability the algorithm outputs a fixed number that is a good approximation to the correct value. Given a time function $T\colon \mathbb{N} \to \mathbb{N}$, we say that an integer-valued function f has a *pseudo-deterministic approximation scheme* (PDAS) running in time T if for each rational number $\epsilon > 0$ there is a function $g\colon \{0,1\}^* \to \mathbb{N}$ and a probabilistic machine M, which given a string x as input, runs in time $T(|x|)$ and outputs $g(x)$ with probability $1-2^{-\Omega(|x|)}$, and moreover we have that $(1-\epsilon)f(x) \leq g(x) \leq (1+\epsilon)f(x)$. A PDAS running in polynomial time is called a PPDAS. We say that an integer-valued function f has a *fully pseudo-deterministic approximation scheme* (FPDAS) running in time T if there is a function $g\colon \{0,1\}^* \times \mathbb{Q}^+ \to \mathbb{N}$ and a probabilistic machine M, which given a string x and a rational number ϵ (in some prespecified format) as input, runs in time $T(|x|, 1/\epsilon)$ and outputs $g(x,\epsilon)$ with probability $1 - 2^{-\Omega(|x|)}$, and moreover we have that $(1-\epsilon)f(x) \leq g(x,\epsilon) \leq (1+\epsilon)f(x)$. An FPDAS running in polynomial time is called a PFPDAS.

Note that a *d*eterministic approximation scheme running in time T is a special case of a PDAS running in time T where the machine M uses no randomness, and similarly a *fully deterministic approximation scheme* running in time T is a special case of an FPDAS running in time T where the machine M uses no randomness.

We also need more relaxed notions of pseudo-deterministic approximation schemes which are not guaranteed to work for all inputs. An *infinitely-often pseudo-deterministic approximation scheme* (i.o.PDAS) is only guaranteed to be pseudo-deterministic and output a correct approximation for infinitely many input lengths (rather than all of them). The notion of *infinitely-often fully pseudo-deterministic approximation scheme* (i.o.FPDAS) is defined analogously.

Finally, given a samplable distribution $\mathfrak{D} = \{\mathcal{D}_n\}$, an i.o.PDAS over $\mathfrak{D}$ is only guaranteed to be pseudo-deterministic and output a correct approximation with probability $1 - 1/n^{\omega(1)}$ over inputs sampled according to $\mathcal{D}_n$, for infinitely many n. Again, the notion of i.o.FPDAS over $\mathfrak{D}$ is defined analogously.

Canonization and approximate canonization. Next we define notions of *canonization* and *approximate canonization* for circuit classes. Let $\mathfrak{C}$ be a circuit class and $s\colon \mathbb{N} \to \mathbb{N}$ be a size function. Given a time function $T\colon \mathbb{N} \to \mathbb{N}$, we say that $\mathfrak{C}(s(n))$ has deterministic (resp. pseudo-deterministic) canonization in time T if there is a deterministic (resp. 1/3-pseudo-deterministic) Turing machine M such that (i) M operates in time $T(n)$ when given as input any circuit C in $\mathfrak{C}(s(n))$, (ii) for any circuit C in $\mathfrak{C}(s(n))$, $M(C)$ is a Boolean circuit

on n variables that is *equivalent* to C, i.e., computes the same Boolean function as C, and (*iii*) for any two equivalent circuits C and C' in $\mathfrak{C}(s(n))$, $M(C) = M(C')$. Note that a pseudo-deterministic canonization algorithm is allowed to output an arbitrary circuit with probability at most $1/3$.

We relax the notion of canonization by requiring the output to only be *approximately* equivalent to the input. Given a parameter $\epsilon > 0$, we say that $\mathfrak{C}(s(n))$ has deterministic (resp. pseudo-deterministic) ϵ-approximate canonization in time T if there is a deterministic (resp. 1/3-pseudo-deterministic) Turing machine M such that (*i*) M operates in time $T(n)$ when given as input any circuit C in $\mathfrak{C}(s(n))$, (*ii*) for any circuit C in $\mathfrak{C}(s(n))$, $M(C)$ is a Boolean circuit on n variables that is an ϵ-approximation to C, i.e., disagrees with C on at most an ϵ-fraction of inputs of length n, and (*iii*) for any two equivalent circuits C and C' in $\mathfrak{C}(s(n))$, $M(C) = M(C')$.

Finally, we stress that in the definition of canonization and approximate canonization it is important that the size bound $s(n)$ is fixed before the formalization of the problem. Indeed, by using an alternative definition that simply postulates that on an arbitrary circuit C from the class $\mathfrak{C}$ the machine M must output (say) in polynomial time on the description length of C an equivalent canonical circuit $M(C)$, one can easily use M to define a *natural property* useful against $\mathfrak{C}$ (in the sense of [30]).[14] However, as far as we know, there might be circuit classes that admit approximate canonization in the original sense introduced above, but do not admit natural properties. The first definition is therefore preferred.

[14] Given a truth-table f, construct an exponential size $\mathfrak{C}$-circuit C for f. Since M is a canonizer and has to run in polynomial time on every circuit D equivalent to C, one can infer from its output on C the approximate $\mathfrak{C}$-circuit complexity of f.

Luby–Veličković–Wigderson Revisited: Improved Correlation Bounds and Pseudorandom Generators for Depth-Two Circuits

Rocco A. Servedio[1]
Department of Computer Science, Columbia University, New York, NY, USA
rocco@cs.columbia.edu

Li-Yang Tan[2]
Department of Computer Science, Stanford University, Stanford, California, USA
liyang@cs.stanford.edu

Abstract

We study correlation bounds and pseudorandom generators for depth-two circuits that consist of a SYM-gate (computing an arbitrary symmetric function) or THR-gate (computing an arbitrary linear threshold function) that is fed by S AND gates. Such circuits were considered in early influential work on unconditional derandomization of Luby, Veličković, and Wigderson [31], who gave the first non-trivial PRG with seed length $2^{O(\sqrt{\log(S/\varepsilon)})}$ that ε-fools these circuits.

In this work we obtain the first strict improvement of [31]'s seed length: we construct a PRG that ε-fools size-S $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AND}$ circuits over $\{0,1\}^n$ with seed length

$$2^{O(\sqrt{\log S})} + \mathrm{polylog}(1/\varepsilon),$$

an exponential (and near-optimal) improvement of the ε-dependence of [31]. The above PRG is actually a special case of a more general PRG which we establish for constant-depth circuits containing multiple SYM or THR gates, including as a special case $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0$ circuits. These more general results strengthen previous results of Viola [47] and essentially strengthen more recent results of Lovett and Srinivasan [30].

Our improved PRGs follow from improved correlation bounds, which are transformed into PRGs via the Nisan–Wigderson "hardness versus randomness" paradigm [37]. The key to our improved correlation bounds is the use of a recent powerful *multi-switching* lemma due to Håstad [21].

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization

Keywords and phrases Pseudorandom generators, correlation bounds, constant-depth circuits

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.56

1 Introduction

Depth-2 circuits which have a SYM or THR gate at the output and AND gates (of arbitrary fan-in) adjacent to the input variables are central objects of interest in concrete complexity, lying at the boundary of our understanding for many benchmark problems such as lower bounds, learning, and pseudorandomness. The class of $\mathsf{SYM} \circ \mathsf{AND}$ circuits has received

[1] Supported by NSF grants CCF-1420349 and CCF-1563155.
[2] Supported by NSF grant CCF-1563122. Part of this research was done during a visit to Columbia University.

much attention even in the restricted case of polylog(n) bottom fan-in (such circuits are also known as SYM^+ circuits) because of the well-known connection with the complexity class ACC^0 [52, 4, 10], a connection that is at the heart of recent breakthrough circuit lower bounds against ACC^0 [50, 34]. Another well-studied subclass, corresponding to the special case where the SYM gate computes the parity of its inputs, is the class of S-sparse polynomials over $\mathbb{F}_2$, which have been intensively studied in a wide range of contexts such as learning [42, 8, 9], approximation and interpolation [26, 18, 40], deterministic approximate counting [16, 27, 31], and property testing [13, 14]. Turning to THR gates (which compute an arbitrary linear threshold function of their inputs) as the top gate, the class of $\mathsf{THR} \circ \mathsf{AND}$ circuits of size-S is easily seen to contain the class of S-sparse polynomial threshold functions over $\{0,1\}^n$. This class, and special cases of it such as low-degree polynomial threshold functions, has also been intensively studied in complexity theory, learning theory, and derandomization, see e.g. [33, 17, 28, 24, 38, 32, 12, 15, 25, 11] and many other works. In this work we focus on *pseudorandom generators* for these $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AND}$ circuits.

In 1993 Luby, Veličković, and Wigderson [31] gave the first pseudorandom generators for these depth-2 circuits. As we shall discuss in detail below, this result was subsequently extended in various ways by different authors, but prior to the present work no strict improvement of Theorem 1 was known for the class of circuits that it addresses.

▶ **Theorem 1** (Luby–Veličković–Wigderson 1993). *There is a PRG with seed length* $2^{O(\sqrt{\log(S/\varepsilon)})}$ *that ε-fools the class of size-S* $\mathsf{SYM} \circ \mathsf{AND}$ *circuits over* $\{0,1\}^n$. *The same is true for the class of size-S* $\mathsf{THR} \circ \mathsf{AND}$ *circuits.*[3]

The main contribution of the present work is an exponential improvment of Theorem 1's dependence on ε, giving the first strict improvement of the [31] seed length:

▶ **Theorem 2** (Our main result). *There is a PRG with seed length* $2^{O(\sqrt{\log S})} + \mathrm{polylog}(1/\varepsilon)$ *that ε-fools the class of size-S* $\mathsf{SYM} \circ \mathsf{AC}^0$ *circuits. The same is true for* $\mathsf{THR} \circ \mathsf{AC}^0$ *circuits.*

Theorem 2 improves on a result of Viola [47] which, improving on [31], gave a $2^{O(\sqrt{\log(S/\varepsilon)})}$-seed-length PRG for size-S $\mathsf{SYM} \circ \mathsf{AC}^0$ circuits. The [47] PRG combines correlation bounds against $\mathsf{SYM} \circ \mathsf{AC}^0$ circuits with the Nisan–Wigderson "hardness versus randomness" paradigm, which yields pseudorandom generators from correlation bounds; we similarly prove Theorem 2 by establishing improved correlation bounds and using the Nisan–Wigderson paradigm.

Near-optimal hardness-to-randomness conversion. A major theme in computational complexity over the the last several decades, dating back to the seminal works of [44, 51, 5, 35, 37], has been that *computational hardness* can be converted into *pseudorandomness*. This insight is at the heart of essentially all unconditional pseudorandom generators, and motivates the goal of understanding when and how this conversion can be carried out in a quantitatively optimal manner. With this perspective in mind, we observe that the dependence on ε in Theorem 2 is optimal up to polynomial factors, and as we discuss in Section 1.3, achieving better dependence on S even for the special case of $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AND}$ circuits would require groundbreaking new lower bounds against low-degree $\mathbb{F}_2$ polynomials and ACC^0 circuits. Hence Theorem 2 achieves a near-optimal hardness-to-randomness conversion for $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0$ circuits; the seed length of our PRG is essentially the best possible given current state-of-the-art correlation bounds and circuit lower bounds.

[3] [31] does not actually consider $\mathsf{THR} \circ \mathsf{AND}$ circuits, but as we discuss later their arguments also apply to this class.

The exponential improvement in $1/\varepsilon$ over [31]'s seed length translates immediately into significantly improved deterministic approximate counting and deterministic search algorithms for $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0$ circuits, two basic algorithmic tasks in unconditional derandomization[4] (see e.g. [2] for formal definitions of these tasks and a discussion of how PRGs yield deterministic algorithms for them).

In the rest of this introduction we provide background and context for our results and explain the main ingredients that underlie them.

1.1 Prior PRGs and correlation bounds for $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0$

As mentioned above, the first results on PRGs for $\mathsf{SYM} \circ \mathsf{AND}$ circuits were given in early influential work of Luby, Veličković, and Wigderson [31], who constructed a PRG that ε-fools size-S $\mathsf{SYM} \circ \mathsf{AND}$ circuits over n variables with seed length $2^{O(\sqrt{\log(S/\varepsilon)})}$. The work of [31] employed ideas similar to those in the "hardness versus randomness" paradigm of [37], which subsequently came to be well understood as a versatile technique for constructing pseudorandom generators from correlation bounds.

A number of years later, with the [37] framework in hand, Viola [47] made the useful observation that correlation bounds against the larger class of $\mathsf{SYM} \circ \mathsf{AND} \circ \mathsf{OR}$ circuits translate to PRGs for $\mathsf{SYM} \circ \mathsf{AND}$ circuits in a "black-box" manner via [37], and the same is true when the top gate is THR instead of SYM. (Informally, the [37] translation "costs" two layers of depth: with typical parameter settings, it yields PRGs for a class $\mathcal{C}$ from correlation bounds against $\mathcal{C} \circ \mathsf{ANY}_{\log n}$ circuits, where an ANY_t gate computes an arbitrary t-variable Boolean function. By rewriting the $\mathsf{ANY}_{\log n}$ gate as a CNF, it is possible to collapse the two adjacent layers of AND gates, yielding Viola's observation.) Roughly speaking, in this translation from correlation bounds against $\mathcal{C} \circ \mathsf{ANY}_{\log n}$ to PRGs that ε-fool $\mathcal{C}$,

- the larger the $\mathcal{C} \circ \mathsf{ANY}_{\log n}$ circuits for which the correlation bound holds, the better (smaller) is the PRG's seed length for fooling size-S functions in $\mathcal{C}$; and
- the smaller the advantage over random guessing that the correlation bound establishes, the better (smaller) is the PRG's seed length's dependence on the fooling parameter ε.

Motivated by this template, [47] established $n^{-\Omega(\log n)}$ correlation bounds against $\mathsf{SYM} \circ \mathsf{AC}^0$ circuits of size $n^{\Omega(\log n)}$. This translates (see Appendix B) into a PRG with seed length $2^{O(\sqrt{\log(S/\varepsilon)})}$ for size-S $\mathsf{SYM} \circ \mathsf{AC}^0$ circuits over $\{0,1\}^n$, matching the seed length achieved by [31] but for a larger class of circuits (and also with a simpler and more modular proof). While [47] does not explicitly discuss THR gates, its proof like that of [31] also goes through for $\mathsf{THR} \circ \mathsf{AC}^0$ as remarked in the earlier footnote.

Subsequent work of [30] established a strong correlation bound of $\exp(-\Omega(n^{1-o(1)}))$ against $\mathsf{SYM} \circ \mathsf{AC}^0$ and a correlation bound of $\exp(-\Omega(n^{1/2-o(1)}))$ against $\mathsf{THR} \circ \mathsf{AC}^0$, but in both cases only for such circuits of size $n^{O(\log\log n)}$. Via the Nisan–Wigderson framework [37] this translates into a PRG with seed length $2^{O(\log S/\log\log S)} + \mathrm{polylog}(1/\varepsilon)$ for size-S $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0$ circuits over $\{0,1\}^n$; while this is a very good dependence on ε, it comes at the cost of a significantly worse dependence on the circuit size S. Thus both the seed length and correlation bounds of [30] are incomparable to those of [31, 47]; see Table 1. We also note that the recent work of Sakai et al. [41] established a range of tradeoffs between circuit size and correlation bounds for (a generalization of) $\mathsf{SYM} \circ \mathsf{AC}^0$ circuits, though for the circuit sizes presented in Table 1 their results are not as strong as the results of [47, 30] listed there.

[4] Indeed, the work of [31] was explicitly motivated by deterministic approximate counting of S-sparse $\mathbb{F}_2$ polynomials; see the abstract of [31].

Table 1 Correlation bounds against $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0{}_d$ circuits and the PRGs that follow via the [37] paradigm. In all cases the "hard function" is the RW function that is defined in (3) and was first considered by Razborov and Wigderson [39]. For a given row, a circuit size of s and a correlation bound of α means that every size-s circuit of the stated type agrees with the n-variable RW function on at most $\frac{1}{2} + \alpha$ fraction of inputs. For each row, see Appendix B for a derivation of how the final column (seed length for a Nisan–Wigderson based PRG) follows from the earlier columns via [37].

	Circuit type	Circuit size S	Correlation bound	PRG seed length
[47]	$\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0{}_d$	$n^{c_d \log n}$	$n^{-c_d \log n}$	$2^{O(\sqrt{\log(S/\varepsilon)})}$
[30]	$\mathsf{SYM} \circ \mathsf{AC}^0{}_d$	$n^{c_d \log\log n}$	$\exp(-n^{1-o(1)})$	$2^{O\left(\frac{\log S}{\log\log S}\right)} + (\log(1/\varepsilon))^{2+o(1)}$
[30]	$\mathsf{THR} \circ \mathsf{AC}^0{}_d$	$n^{c_d \log\log n}$	$\exp(-n^{1/2-o(1)})$	$2^{O\left(\frac{\log S}{\log\log S}\right)} + (\log(1/\varepsilon))^{4+o(1)}$
This work	$\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0{}_d$	$n^{c \log n}$	$\exp(-\Omega(n^{0.499}))$	$2^{O(\sqrt{\log S})} + (\log(1/\varepsilon))^{4.01}$

(We further note that other incomparable results have been achieved in separate lines of work on pseudorandom generators for degree-d polynomial threshold functions [12, 32, 25] and degree-d $\mathbb{F}_2$ polynomials [6, 7, 29, 49], which correspond to $\mathsf{THR} \circ \mathsf{AND}_d$ and $\mathsf{PAR} \circ \mathsf{AND}_d$ circuits respectively. The seed lengths of these PRGs all have an exponential dependence on d, and thus do not yield non-trivial results for general $\mathrm{poly}(n)$-size $\mathsf{THR} \circ \mathsf{AND}$ or $\mathsf{PAR} \circ \mathsf{AND}$ circuits. For constant d, the [29, 49] PRGs for $\mathsf{PAR} \circ \mathsf{AND}_d$ circuits achieve optimal seed length, while the [32, 25] PRGs for $\mathsf{THR} \circ \mathsf{AND}_d$ have seed length $\mathrm{poly}(1/\varepsilon) \cdot \log n$.)

1.2 Our main technical contribution: New correlation bounds against $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0$ circuits

The technical heart of our main result is a new exponential correlation bound against $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0$ circuits of size $n^{\Omega(\log n)}$:

▶ **Theorem 3.** *There is an absolute constant $\tau > 0$ and an explicit $\mathrm{poly}(n)$-time computable function $H : \{0,1\}^n \to \{0,1\}$ with the following property: for any constant d, for n sufficiently large it is the case that for any n-variable circuit C of size $n^{\tau \log n}$ and depth d with a SYM or THR gate at the top, we have*

$$\Pr_{\boldsymbol{x} \leftarrow \{0,1\}^n}[H(\boldsymbol{x}) = C(\boldsymbol{x})] \leq \frac{1}{2} + \exp(-\Omega(n^{0.499})).$$

Theorem 3 strictly improves on the correlation bound provided by Theorem 4 of [47], as it establishes correlation bounds for the same class of $n^{\Omega(\log n)}$-size circuits, but gives a much smaller $\exp(-\Omega(n^{0.499}))$ upper bound on the correlation rather than $n^{-\Omega(\log n)}$. As described in Appendix B, our PRG result for $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0$ (Theorem 2) follows directly from Theorem 3 via the Nisan–Wigderson framework. In Section 1.4 we give an overview of the ideas that underlie our new correlation bound.

Correlation bounds and PRGs for constant-depth circuits with multiple SYM or THR gates. The main correlation bound and PRG of [47] are actually for $n^{c_d \log n}$-size depth-d circuits with $c_d(\log n)^2$ many SYM gates, and similarly the main result of [30] is a correlation bound for constant-depth circuits with $n^{1-o(1)}$ many SYM gates or $n^{1/2-o(1)}$ many THR gates. Our results similarly extend to constant-depth circuits with multiple SYM or THR gates. Our most general correlation bound is the following:

▶ **Theorem 4.** *There is an absolute constant $\tau > 0$ and an explicit* $\mathrm{poly}(n)$*-time computable function $H : \{0,1\}^n \to \{0,1\}$ with the following property: for any constant d, for n sufficiently large, any n-variable circuit C of size $n^{\tau \log n}$ and depth d containing $n^{0.249}$ many* SYM *or* THR *gates (the circuit is allowed to contain both types of gates) satisfies*

$$\Pr_{\boldsymbol{x} \leftarrow \{0,1\}^n}[H(\boldsymbol{x}) = C(\boldsymbol{x})] \leq \frac{1}{2} + \exp(-\Omega(n^{0.249})).$$

Via the Nisan–Wigderson framework, Theorem 4 immediately yields the following, which is our most general PRG result:

▶ **Corollary 5.** *For some sufficiently small absolute constant $c > 0$, there is a PRG with seed length $2^{O(\sqrt{\log S})} + \mathrm{polylog}(1/\varepsilon)$ that ε-fools the class of size-S constant-depth circuits that contain $2^{\sqrt{c \log S}}$ many* SYM *or* THR *gates.*

This strictly improves the main [47] PRG (Theorem 1 of [47]), which achieves seed length $2^{O(\sqrt{\log(S/\varepsilon)})}$ for size-S constant-depth circuits that contain $O((\log S)^2)$ many SYM or THR gates. We prove Theorem 4 in Appendix C.

1.3 Barriers to further progress: correlation bounds for $\mathbb{F}_2$ polynomials and ACC^0 lower bounds

In this section we outline why achieving better dependence on S will require groundbreaking new correlation bounds or circuit lower bounds.

The seminal work of Babai, Nisan, and Szegedy [3] gave an explicit function and established that it has exponentially small correlation $\exp(-\Omega(n/4^d\, d))$ with any n-variable $\mathbb{F}_2$ polynomials of degree d (see Theorem 3 of [48]). This result (and the multiparty communication-based techniques underlying it) have had far-reaching consequences in complexity theory; 25 years later, improving on this correlation bound remains a prominent open problem. In particular, even achieving correlation bounds of the form $\frac{1}{2} + n^{-1}$ against polynomials of degree $\log n$ with respect to any explicit distribution $\mathcal{D}$ would constitute a significant breakthrough (see e.g. "Open Question 1" in Viola's excellent survey [48]). Since every degree-d polynomial is s-sparse for $s = \binom{n}{d}$, this is clearly a special case of obtaining $\frac{1}{2} + n^{-1}$ correlation bounds against polynomials of sparsity $s = \binom{n}{\log n}$. Via a standard connection between PRGs and correlation bounds (see e.g. Proposition 3.1 of [49]), an improvement in the dependence on S in Theorem 2 to $2^{o(\sqrt{\log S})} + \mathrm{polylog}(1/\varepsilon)$, even for the special case of S-sparse $\mathbb{F}_2$ polynomials, would immediately yield exponentially-small correlation bounds against s-sparse $\mathbb{F}_2$ polynomials for $s = n^{\omega(\log n)} \gg \binom{n}{\log n}$ with respect to an explicit distribution. (We remark that the same correlation bounds are also open for the class of degree $\log n$ polynomial threshold functions, and hence the same barrier applies to improving the S-dependence of PRGs for size-S $\mathsf{THR} \circ \mathsf{AND}$ circuits.)

Further improvements of Theorem 2 would have even more dramatic consequences. Classical "depth-compression" results of Yao [52] and Beigel and Tauri [4] (see also [10]) show that every size-s depth-d ACC^0 circuit can be computed by a size-S $\mathsf{SYM} \circ \mathsf{AND}$ circuit where $S = \exp((\log n)^{O_d(1)})$. Improving the seed length of Theorem 2 for the class of size-S $\mathsf{SYM} \circ \mathsf{AND}$ circuits to $2^{(\log S)^{o(1)}}$ (even for constant ε) would therefore separate NP from ACC^0, a significant strengthening of Williams's celebrated separation of NEXP from ACC^0 [50].

1.4 The high-level structure of our correlation bound argument

We recall the "bottom-up" approach to proving correlation bounds via the method of random restrictions. This approach dates back to the classic correlation bounds between Parity

and AC^0 of [1] and [20]; in particular, the relevant prior works of [47, 30] also operate within this framework.

Fix a hard function H, and let F be any function belonging to a given class $\mathcal{F}$ of Boolean functions (in our case $\mathcal{F}$ is the class of $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0$ circuits of size $n^{\Omega(\log n)}$). Our goal is to show that F has small *correlation* with H, i.e. that $\Pr_{\boldsymbol{x} \leftarrow \{0,1\}^n}[F(\boldsymbol{x}) = H(\boldsymbol{x})] \leq \frac{1}{2} + \alpha$ for some small α where $\boldsymbol{x}$ is uniform over $\{0,1\}^n$. This can be achieved by designing a *fair* distribution $\mathcal{R}$ over random restrictions that satisfies the following two competing requirements. (A distribution $\mathcal{R}$ over restrictions is said to be fair if first drawing a restriction $\boldsymbol{\rho} \leftarrow \mathcal{R}$ and then filling in all $*$'s to independent uniform values from $\{0,1\}$ results in a uniform random string from $\{0,1\}^n$.)

(1) Approximator (F) simplifies: With high probability $1 - \gamma_{\mathrm{SL}}$ over $\boldsymbol{\rho} \leftarrow \mathcal{R}$, F "collapses" when it is hit by $\boldsymbol{\rho}$, meaning that $F \restriction \boldsymbol{\rho} \in \mathcal{F}_{\mathrm{simple}}$ for some class $\mathcal{F}_{\mathrm{simple}} \subseteq \mathcal{F}$. Looking ahead, in our case

$$\mathcal{F}_{\mathrm{simple}} = \left\{ \{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AND}_{k:=0.0005 \log m} \text{ circuits} \right\}$$

where $m \approx \sqrt{n}$ and AND_k denotes the class of fan-in k AND gates. A collapse to this $\mathcal{F}_{\mathrm{simple}}$ is useful for us because there are efficient multiparty communication protocols for functions computable by $\mathcal{F}_{\mathrm{simple}}$ (due to [22] when the top gate is SYM and to [36] when it is THR).

(2) Target (H) retains structure: With high probability $1 - \gamma_{\mathrm{target}}$ over $\boldsymbol{\rho} \leftarrow \mathcal{R}$, the restricted hard function $H \restriction \boldsymbol{\rho}$ "retains structure", in the sense that it has small correlation with every function in $\mathcal{F}_{\mathrm{simple}}$. In our case our notion of structure will be that $H \restriction \boldsymbol{\rho}$ "contains a perfect copy of" the generalized inner product function:

$$\mathrm{GIP}_{m/2,k+1}(x) := \bigoplus_{i=1}^{m/2} \bigwedge_{j=1}^{k+1} x_{i,j},$$

where m and k are the same m and k as above.

Suppose we have such a fair distribution $\mathcal{R}$ over random restrictions satisfying (1) and (2) above. The remaining step in the argument is to show the following: (3) for any ρ such that both of the above happen (approximator simplifies and the hard function retains structure), $F \restriction \rho$ and $H \restriction \rho$ have small correlation, i.e. they agree on at most $\frac{1}{2} + \gamma_{\mathrm{corr}}$ fraction of all inputs. As in the previous works of [47, 30], the fact that $\mathcal{F}_{\mathrm{simple}}$ and $\mathrm{GIP}_{m/2,k+1}$ have small correlation follows from a celebrated theorem of Babai, Nisan, and Szegedy [3] lower bounding the multiparty communication complexity of $\mathrm{GIP}_{m,k+1}$.

It is straightforward to see that items (1)–(3) above establish a correlation bound of

$$\Pr_{\boldsymbol{x} \leftarrow \{0,1\}^n}[\, F(\boldsymbol{x}) = H(\boldsymbol{x}) \,] \leq \frac{1}{2} + \gamma_{\mathrm{SL}} + \gamma_{\mathrm{target}} + \gamma_{\mathrm{corr}}.$$

The goal is therefore to carry out the above with $\max\{\gamma_{\mathrm{SL}}, \gamma_{\mathrm{target}}, \gamma_{\mathrm{corr}}\}$ as small as possible for the class $\mathcal{F}$ of $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0{}_d$ circuits of as large a size as possible. As indicated earlier, for any constant d and circuits of size up to $s = n^{\tau \log n}$ we achieve $\max\{\gamma_{\mathrm{SL}}, \gamma_{\mathrm{target}}, \gamma_{\mathrm{corr}}\} = \exp(-\Omega(n^{0.499}))$.

We upper bound γ_{SL}, γ_{target}, and γ_{corr} in Sections 2, 3, and 4 respectively. Some useful preliminaries are given in Appendix A.

1.5 How this work differs from [47, 30]: improved depth reduction

A simple observation (due to [19]) that is used in both [47, 30] and in our work as well is the fact that a symmetric function of depth-k decision trees can be simulated by a (different) symmetric function of width-k AND's, and likewise for a threshold function of depth-k decision trees. (See Fact 1 for a precise statement.) Consequently we can think of $\mathcal{F}_{\text{simple}}$ as $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{DT}_k$ rather than $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AND}_k$ (where DT_k denotes the class of decision trees of depth k), and for depth reduction it suffices to prove that a family of s many AC^0 circuits collapses to a family of small-depth decision trees with high probability under a random restriction. This is exactly what is shown by *switching lemmas*.

The loss in the previous works of [47, 30] is due to the switching lemmas they use and the limitations of these switching lemmas. [47] uses the standard [20] switching lemma:

▶ **Theorem 6** (Håstad's switching lemma). *Let F be computed by a depth-2 circuit with bottom fan-in w. Then*

$$\Pr_{\boldsymbol{\rho} \leftarrow \mathcal{R}_p} \left[F \restriction \boldsymbol{\rho} \text{ is not a depth-}t \text{ decision tree} \right] \leq (5pw)^t.$$

This failure probability of $(5pw)^t$ cannot be made exponentially small in our setting: since correlation bounds strong enough to be useful for the [37] framework are not known for $\mathsf{SYM} \circ \mathsf{AND}_{\omega(\log n)}$ (see "Open Question 1" of [48]) the value of t has to has to be taken to be at most $k = O(\log n)$, and moreover p certainly has to be $\gg 1/n$ (since taking $p = 1/n$ would leave only a constant number of coordinates alive, and $H \restriction \boldsymbol{\rho}$ would not "retain structure" in the sense of containing a copy of $\mathrm{GIP}_{m/2,k+1}$). Indeed, [47] applies Theorem 6 with $p = n^{-\Theta(1)}$ in order to make the failure probability as small as $n^{-\Omega(\log n)}$, and this is why [47] only achieves quasi-polynomial correlation bounds $n^{-\Omega(\log n)}$.

Faced with this obstacle, instead of using the standard [20] switching lemma, [30] reverts to the earlier "multi-switching lemma" of [1] which applies to a collection of depth-2 circuits rather than a single such circuit. The [1] multi-switching lemma, stated below, *does* achieve exponentially small failure probability, but is only able to handle collections of $n^{O(\log \log n)}$ many k-DNFs, for $k = O(\log \log n)$. Recall that a restriction tree T is like a decision tree except that leaves do not have labels associated with them (so each root-to-leaf path is a restriction). The distribution μ_T corresponds to the distribution over restrictions obtained by making a random walk from the root of T.

▶ **Theorem 7** (Ajtai's switching lemma [1]). *Let $\mathcal{F} = \{F_1, \ldots, F_s\}$ be a family of s many DNFs over $x_1, \ldots, x_n$, each of width k. For any $t \geq 1$, there is a restriction tree T of height at most $nk(\log s)/(\log n)^t$ such that*

$$\Pr_{\boldsymbol{\rho} \leftarrow \mu_T} \left[F_i \restriction \boldsymbol{\rho} \text{ is not a } (\log n)^{10kt2^k}\text{-junta} \right] \leq 2^{-n/(2^{10k}(\log n)^t)}.$$

Hence [30] achieves exponentially small correlation bounds (the main point of their paper), but only against circuits of size $n^{O(\log \log n)}$.

The key new ingredient that we employ in this work is a recent powerful multi-switching lemma from [21]. (We note that [23] gives an essentially equivalent multi-switching lemma which we could also use.) Roughly speaking the [21] multi-switching lemma, whose precise statement we defer to Section 2 as it is somewhat involved, lets us achieve an exponentially small failure probability (like Ajtai's multi-switching lemma) of achieving a significantly more drastic simplification than Ajtai's multi-switching lemma (recall the doubly-exponential-in-k dependence on the junta size in Theorem 7). This quantitative improvement in depth reduction translates into our stronger correlation bounds.

1.6 Relation to [43]

We close this introduction by discussing the connection between this paper and recent concurrent work of the authors [43]. The high-level approaches of the two papers are fairly different: unlike the current paper, [43] does not use the Nisan–Wigderson hardness-versus-randomness paradigm (and does not establish any new correlation bounds); instead it establishes a derandomized version of the [21] multi-switching lemma and combines this with other ingredients to obtain its final PRG in a manner reminiscent of [2, 45].

The results of the two papers are also incomparable (briefly, [43] obtains significantly shorter seed length for significantly more restricted classes of functions). The first main result of [43] is an ε-PRG for the class of size-S depth-d AC^0 circuits with seed length $\log(S)^{d+O(1)} \cdot \log(1/\varepsilon)$. This is incomparable to the most closely related result of the present paper (Corollary 5, which gives a $2^{O(\sqrt{\log S})} + \mathrm{polylog}(1/\varepsilon)$ seed length PRG for AC^0 circuits augmented with polynomially many SYM or THR gates), since the [43] result gives a significantly better seed length but for the significantly more limited class of "un-augmented" constant-depth circuits (indeed, the [43] result does not apply to AC^0 circuits augmented even with a single SYM or THR gate). The second main result of [43] is an ε-PRG for the class of S-sparse $\mathbb{F}_2$ polynomials with seed length $2^{O(\sqrt{\log S})} \cdot \log(1/\varepsilon)$. Here too the seed length of [43] is shorter than that of the current paper (giving the optimal $\log(1/\varepsilon)$ dependence on ε as opposed to the $(\log(1/\varepsilon))^{4.01}$ of the current paper), but the result of [43] only holds for S-sparse $\mathbb{F}_2$ polynomials, which are a very restricted case of the $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0{}_d$ circuits which are handled in the current paper.

2 Ingredient (1): Simplifying the approximator

The main result of this section is the following:

▶ **Lemma 8.** *Let F be any $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0{}_d$ circuit of size $s = n^{\tau \log n}$. There is a fair distribution $\mathcal{R}$ over restrictions $\boldsymbol{\rho} \in \{0, 1, *\}^n$ such that the following holds: With probability $1 - \gamma_{\mathrm{SL}} = 1 - \exp(-\Omega_d(\sqrt{n/\log n}))$ over the draw of $\boldsymbol{\rho} \leftarrow \mathcal{R}$, it is the case that $F \restriction \boldsymbol{\rho}$ belongs to the class $\mathcal{F}_{simple} = \{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AND}_{k:=0.0005 \log m}$, where $m = \Theta(\sqrt{n/\log n})$.*

The recent "multi-switching lemma" of [21] is the main technical tool we use to establish Lemma 8. To state the [21] lemma we need some terminology. Let $\mathcal{G}$ be a family of Boolean functions. A restriction tree T is said to be a *common ℓ-partial restriction tree (RT) for* $\mathcal{G}$ if every $g \in \mathcal{G}$ can be expressed as T with depth-ℓ decision trees hanging off its leaves. (Equivalently, for every $g \in \mathcal{G}$ and root-to-leaf path π in T, we have that $g \restriction \pi$ is computed by a depth-ℓ decision tree.)

▶ **Theorem 9** ([21] multi-switching lemma)**.** *Let $\mathcal{F} = \{F_1, \ldots, F_s\}$ be a collection of depth-2 circuits with bottom fan-in w. Then for any $t \geq 1$,*

$$\Pr_{\boldsymbol{\rho}' \leftarrow \mathcal{R}_p} \left[\mathcal{F} \restriction \boldsymbol{\rho}' \text{ does not have a common } (\log s)\text{-partial RT of depth} \leq t \right] \leq s(24pw)^t.$$

Theorem 9 is the main tool we use to simplify any $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0$ circuit down to an $\mathcal{F}_{\mathrm{simple}}$-circuit. Conceptually, we think of this transformation as being done in three steps:

1. (Main step) Apply a random restriction $\boldsymbol{\rho}' \leftarrow \mathcal{R}_p$ to convert a $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0$ circuit into a decision tree with a $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{DT}$ circuit at each leaf.
2. Observing that $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{DT} \equiv \{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AND}$, this is equivalent to a decision tree with a $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AND}$ circuit at each leaf.

3. Trim the fan-in of the AND gates in each $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AND}$ circuit (by increasing the depth of the decision tree). The last step in the draw of a random restriction $\boldsymbol{\rho}$ from the overall fair distribution $\mathcal{R}$ corresponds to a random walk down this final decision tree.

In the rest of this section we describe each of these steps in detail and thereby prove Lemma 8.

First (main) step. If g is a Boolean function and $\mathcal{C}$ is a class of circuits, we say that g is *computed by a $(d, \mathcal{C})$-decision tree* if g is computed by a decision tree of depth d (with a single Boolean variable at each internal node as usual) in which each leaf is labeled by a function from $\mathcal{C}$. We require the following corollary of Theorem 9:

▶ **Corollary 10.** *Let G be any Boolean function and G be a gate computing G, and let F be a $\mathsf{G} \circ \mathsf{AC}^0{}_d$ circuit of size s . Then for $p = \frac{1}{48}(48 \log s)^{-(d-1)}$ and any $t \geq 1$,*

$$\Pr_{\boldsymbol{\rho}' \leftarrow \mathcal{R}_p} \left[F \restriction \boldsymbol{\rho}' \text{ is not computed by a } (2^d t, \mathsf{G} \circ \mathsf{DT}_{\log s})\text{-decision tree} \right] \leq s \cdot 2^{-t}.$$

Proof. We may assume without loss of generality that the depth-$(d+1)$ circuit F is *layered*, meaning that for any gate g it contains, every directed path from an input variable to g has the same length (converting an unlayered circuit to a layered one increases its size only by a factor of d, which is negligible for our purposes). Let s_i denote the number of gates in layer i (at distance i from the inputs), so $s = s_1 + \cdots + s_d$.

We begin by trimming the bottom fan-in of F: applying Theorem 9 with $\mathcal{F}$ being the s_1 many bottom layer gates of F (viewed as depth-2 circuits of bottom fan-in $w = 1$) and $p_0 := 1/48$, we get that

$$\Pr_{\boldsymbol{\rho}_0 \leftarrow \mathcal{R}_{p_0}} \big[F \restriction \boldsymbol{\rho}_0 \text{ is not computed by a } (t, G \circ \mathsf{AC}^0(\text{depth } d, \text{ bottom fan-in } \log s))\text{-decision tree} \big] \leq s_1 \cdot 2^{-t}.$$

Let $F^{(0)}$ be any good outcome of the above, a $(t, G \circ \mathsf{AC}^0(\text{depth } d, \text{ bottom fan-in } \log s))$-decision tree. Note that there are at most 2^t many $\mathsf{AC}^0(\text{depth } d, \text{ fan-in } \log s)$ circuits at the leaves of the depth-t decision tree. Applying Theorem 9 to each of them with $p_1 := 1/(48 \log s)$ (and the 't' of Theorem 9 being $2t$) and taking a union bound over all 2^t many of them, we get that

$$\Pr_{\boldsymbol{\rho}_1 \leftarrow \mathcal{R}_{p_1}} \big[F^{(0)} \restriction \boldsymbol{\rho}_1 \text{ is not a } (t + 2t, G \circ \mathsf{AC}^0(\text{depth } d-1, \text{ fan-in } \log s))\text{-decision tree} \big] \leq s_2 \cdot 2^{-2t} \cdot 2^t = s_2 \cdot 2^{-t}.$$

Repeat with $p_2 = \ldots = p_{d-1} := 1/(48 \log s)$, each time invoking Theorem 9 with its 't' being the one more than the current depth of the decision tree . The claim then follows by summing the $s_1 2^{-t}$, $s_2 2^{-t}, \ldots, s_d 2^{-t}$ failure probabilities over all d stages and the fact that

$$\prod_{j=0}^{d-1} p_i = \frac{1}{48} \cdot \frac{1}{(48 \log s)^{d-1}} = p. \qquad \blacktriangleleft$$

Second step: From $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{DT}$ to $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AND}$. We recall the following fact from [19]:

▶ **Fact 1.** *Every $\mathsf{SYM}_s \circ \mathsf{DT}_{\log s}$ function (resp. $\mathsf{THR}_s \circ \mathsf{DT}_{\log s}$) can be computed by a $\mathsf{SYM}_{s^2} \circ \mathsf{AND}_{\log s}$ (resp. $\mathsf{THR}_{s^2} \circ \mathsf{AND}_{\log s}$) circuit.*

(This is an easy consequence of the fact that any decision tree may be viewed as a DNF whose terms corresponds to the paths to 1-leaves, and that this DNF has the property that any input assignment makes at most one term true.) Applying Fact 1 and choosing $t = m/2^{d+1}$ in Corollary 10, (where $m = \Theta(\sqrt{n/\log n})$ will be defined precisely in the next section), we get the following special case of Corollary 10:

▶ **Corollary 11.** *Let F be a $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0{}_d$ circuit of size $s = n^{\tau \log n}$. Then for $p = \frac{1}{48}(48 \log s)^{-(d-1)}$,*

$$\begin{aligned}
&\underset{\boldsymbol{\rho}' \leftarrow \mathcal{R}_p}{\boldsymbol{Pr}} \left[F \restriction \boldsymbol{\rho}' \text{ is not computed by a } (m/2, \{\mathsf{SYM}_{s^2}, \mathsf{THR}_{s^2}\} \circ \mathsf{AND}_{\log s})\text{-decision tree} \right] \\
&\leq s \cdot 2^{-t} = s \cdot 2^{-m/2^{d+2}} \\
&= \exp(-\Omega_d(\sqrt{n/\log n})) \;:=\; \gamma_{\mathrm{SL}}. \qquad (1)
\end{aligned}$$

Third step: Trimming to reduce bottom fan-in. The $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AND}$ circuits hanging off the leaves of our decision tree have bottom fan-in at most $\log s$, but we will need them to have fan-in at most k in order to invoke the [3] lower bound later. At each leaf ℓ we achieve this smaller fan-in by identifying a set (call it S_ℓ) of additional variables and restricting them in all possible ways; we argue that every fixing of the variables in S_ℓ gives the desired upper bound of k on the bottom-AND fan-in. We use a probabilistic argument to establish the existence of the desired set S_ℓ (this is important because in the next section we will need each S_ℓ to satisfy an additional property, and the probabilistic argument makes it easy to achieve this).

Let us write "$\boldsymbol{L} \subseteq_q X$" to indicate that $\boldsymbol{L}$ is a subset of X that is randomly chosen by independently including each element of X with probability q. We will use the following easy result:

▶ **Fact 2.** *Let $\{C_1, \ldots, C_{s^2}\}$ be a collection of subsets of $[n]$ where each $|C_i| \leq w$. Then for $\boldsymbol{L} \subseteq_q [n]$ and $k \leq w$, we have*

$$\underset{\boldsymbol{L} \subseteq_q [n]}{\boldsymbol{Pr}} \left[\exists\, i \in [s^2] \text{ such that } |C_i \cap \boldsymbol{L}| > k \right] \leq s^2 \binom{w}{k} q^k.$$

Recall that $s = n^{\tau \log n}$ where $\tau > 0$ is a small absolute constant to be specified later and that $k = 0.0005 \log n$. We set

$$q := \frac{k}{e \log s} \cdot 2^{-(3 \log s)/k} = \frac{1}{\Theta(\log n)} \cdot n^{-\Theta(1)\cdot\tau} < n^{-0.01},$$

where the last inequality holds for a suitably small choice of the constant τ. Observe that q is chosen so as to ensure

$$s^2 \binom{\log s}{k} q^k \leq 2^{2 \log s} \left(\frac{e \log s}{k} \cdot q \right)^k = \frac{1}{s} \ll 1. \qquad (2)$$

Fix T to be an $(m/2, \{\mathsf{SYM}_{s^2}, \mathsf{THR}_{s^2}\} \circ \mathsf{AND}_{\log s})$-decision tree as given by Corollary 11. At each leaf ℓ of T, draw a set $\boldsymbol{L}(\ell) \subseteq_q [n]$ and let $\mathbf{S}_\ell$ be $([n] \setminus \mathrm{fixed}(\ell)) \setminus \boldsymbol{L}(\ell)$, where $\mathrm{fixed}(\ell) \subseteq [n]$ is the subset of variables that are fixed on the root-to-ℓ path in T. By Fact 2 and (2), at each leaf ℓ it is the case that with probability at least $1 - 1/s$ over the random draw of $\boldsymbol{L}(\ell)$, every extension of the root-to-ℓ path in T that additionally fixes all the variables in $\mathbf{S}_\ell$ collapses the $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AND}_{\log s}$ circuit that was at ℓ in T down to a

$\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AND}_k$ circuit. We say that such an outcome of $\boldsymbol{L}(\ell)$ is a *good* outcome (we will refer back to this notion in the next section).

In summary, the above discussion establishes Lemma 8, where the fair distribution $\mathcal{R}$ corresponds to

(a) first drawing $\boldsymbol{\rho}' \leftarrow \mathcal{R}_p$,

(b) then walking down a random root-to-leaf path π in the resulting depth-$(m/2)$ decision tree given by Corollary 11,

(c) and then finally, at the resulting leaf ℓ, choosing a random assignment to the variables in the set S_ℓ that corresponds to $L(\ell)$, where $L(\ell)$ is a good outcome of the random variable $\boldsymbol{L}(\ell) \subseteq_q [n]$. (Note that the randomness over $\boldsymbol{L}(\ell)$ is not part of the random draw of $\boldsymbol{\rho} \leftarrow \mathcal{R}$; all we require is the existence of a good $L(\ell)$.)

Based on our discussion thus far each $L(\ell)$ may be fixed to be any good outcome of $\boldsymbol{L}(\ell)$; we will give an additional stipulation on $L(\ell)$ in Remark 3.

3 Ingredient (2) (target retains structure): GIP ∘ PAR under random restrictions

Like [47, 30], our hard function will be the generalized inner product function composed with parity:

$$\mathrm{RW}_{m,k,r}(x) = \bigoplus_{i=1}^{m} \bigwedge_{j=1}^{k+1} \bigoplus_{\ell=1}^{r} x_{i,j,\ell}. \tag{3}$$

This function was introduced by Razborov and Wigderson [39] to show $n^{\Omega(\log n)}$ lower bounds against depth-3 threshold circuits with AND gates at the bottom layer. We will set

$$m = r = \sqrt{n/(k+1)} \quad \text{(recall that } k = 0.0005 \log m\text{)}.$$

Note that $m = r = \Theta(\sqrt{n/\log n})$ and $k = \Theta(\log n)$. Given parameters m', k', r', we say that a function $g : \{0,1\}^n \to \{0,1\}$ *contains a perfect copy of* $\mathrm{RW}_{m',k',r'}$ if there is a restriction κ such that $(g \restriction \kappa)(x) = b \oplus \bigoplus_{i=1}^{m'} \bigwedge_{j=1}^{k+1} \left(b_{i,j} \bigoplus_{\ell=1}^{r'} x_{i,j,k}\right)$ for some bits $b, b_{i,j}$.

Roughly speaking, the motivation behind augmenting GIP with a layer of parities is to ensure that RW is resilient to random restrictions (i.e. that RW $\restriction \boldsymbol{\rho}$ "remains complex", containing a copy of GIP with high probability after a suitable random restriction). In our setting we need that RW is resilient to a random restriction $\boldsymbol{\rho} \leftarrow \mathcal{R}$ for the fair distribution $\mathcal{R}$ from Lemma 8; we establish this in the rest of this section.

▶ **Proposition 1.** *Consider the space of formal variables of* $\mathrm{RW}_{m,k,r} : \{0,1\}^n \to \{0,1\}$*:*

$$X = \left\{x_{i,j,t} \colon (i,j,t) \in [m] \times [k+1] \times [r]\right\}, \quad |X| = m(k+1)r := n.$$

Then for $p = \frac{1}{48}(48 \log s)^{-(d-1)}$ *(as in Corollary 11),*

$$\Pr_{\boldsymbol{L} \subseteq_p X}\left[\exists\,(i,j) \colon \left|\{t \in [r] \colon x_{i,j,t} \in \boldsymbol{L}\}\right| < \frac{pr}{2}\right] \le m(k+1) \cdot \exp\left(-\Omega(pr)\right).$$

Proof. This follows directly from a standard multiplicative Chernoff bound and a union bound over all $(i,j) \in [m] \times [k+1]$. ◀

Recall that a random restriction $\boldsymbol{\rho}' \leftarrow \mathcal{R}_p$ can be thought of as being sampled by first drawing $\mathbf{K} \subseteq_p X$ and setting $\boldsymbol{\rho}_i$ to $*$ for each $i \in \mathbf{K}$, and then setting the coordinates of $\boldsymbol{\rho}'$ in $X \setminus \mathbf{K}$ according to a uniform random draw from $\{0,1\}^{X \setminus \mathbf{K}}$. Proposition 1 and the definition of RW thus yield the following:

▶ **Corollary 12.** *For $\boldsymbol{\rho}' \leftarrow \mathcal{R}_p$, for $p = \frac{1}{48}(48 \log s)^{-(d-1)}$, $\mathrm{RW}_{m,k,r}(x) \restriction \boldsymbol{\rho}'$ contains a perfect copy of*

$$\mathrm{RW}_{m,k,r'}(x) = \bigoplus_{i=1}^{m} \bigwedge_{j=1}^{k+1} \bigoplus_{t=1}^{r'} x_{i,j,t}, \qquad \textit{where} \quad r' = \frac{pr}{2}$$

with failure probability at most

$$\exp(-\Omega(pr)) = \exp\left(-\frac{\sqrt{n/\log n}}{(\Theta(\log s))^{d-1}}\right) := \gamma_{\mathrm{target}}. \tag{4}$$

Note that

$$r' = \frac{pr}{2} = \frac{\sqrt{n/\log n}}{(\Theta(\log s))^{d-1}} > n^{0.49},$$

where the inequality uses the fact that d is a constant and the fact that $s = n^{O(\log n)}$; we will use this later.

Corollary 12 states that with very high probability over $\boldsymbol{\rho}' \leftarrow \mathcal{R}_p$, the function $\mathrm{RW}_{m,k,r} \restriction \boldsymbol{\rho}'$ "does not simplify too much"; however we need $\mathrm{RW}_{m,k,r}$ to "not simplify too much" under a full random restriction drawn from $\mathcal{R}$ (recall the discussion at the end of Section 2). We proceed to establish this.

Fix any outcome ρ' of $\boldsymbol{\rho}' \leftarrow \mathcal{R}_p$ such that (i) the conclusion of Corollary 11 holds (i.e. $F \restriction \rho'$ is computed by a $(m/2, \{\mathsf{SYM}_{s^2}, \mathsf{THR}_{s^2}\} \circ \mathsf{DT}_{\log s})$-decision tree, which we call T), and (ii) the conclusion of Corollary 12 holds (i.e. $\mathrm{RW}_{m,k,r} \restriction \rho'$ contains a perfect copy of $\mathrm{RW}_{m,k,r'}$). (A random $\boldsymbol{\rho}' \leftarrow \mathcal{R}_p$ is such an outcome with probability at least $1 - \gamma_{\mathrm{SL}} - \gamma_{\mathrm{target}}$.) For ease of notation let us write RW' to denote $\mathrm{RW}_{m,k,r} \restriction \rho'$.

Fix any path π that reaches a leaf ℓ in T. (Note that a random choice of such a path corresponds to part (b) in the random draw of $\boldsymbol{\rho} \leftarrow \mathcal{R}$, recalling the discussion at the end of Section 2.) Since $|\pi| \leq m/2$, we have that the set

$$A_\ell := \{i \in [m] \colon \pi_{i,j,t} = * \text{ for all } j \in [k+1] \text{ and all } t\}$$

has cardinality at least $m - |\pi| \geq m/2$. In words, at least $m/2$ of the m many depth-2 subcircuits of RW' are completely "untouched" by π. For part (c) of the draw from $\mathcal{R}$, recall that the set $L(\ell)$ could be taken to be any good outcome of $\boldsymbol{L}(\ell)$, and that a random $\boldsymbol{L}(\ell) \subseteq_q [n]$ is good with probability at least $1 - 1/s$. By the same Chernoff bound argument as the one in Proposition 1, we have that

$$\begin{aligned}\Pr_{\boldsymbol{L} \subseteq_q [n]}\left[\exists\,(i,j) \in A_\ell \times [k+1] \colon \left|\{t \colon x_{i,j,t} \in \boldsymbol{L}(\ell)\}\right| < \frac{qr'}{2}\right] &\leq |A_\ell|(k+1)\exp\left(-\Omega(qr')\right) \\ &\ll \exp(-\Omega(n^{0.48})),\end{aligned}$$

recalling that $|A_\ell| \leq m$, $k = \Theta(\log n)$, $q \geq n^{-0.01}$ and $r' > n^{0.49}$. Since $1 - 1/s + 1 - \exp(-\Omega(n^{0.48})) > 1$, there must exist a good outcome $L(\ell)$ of $\boldsymbol{L}(\ell)$ such that for the

corresponding S_ℓ, every restriction ρ^{trim} fixing precisely the variables in S_ℓ is such that $\text{RW}' \restriction \pi\rho^{\text{trim}}$ contains a perfect copy of

$$\text{RW}_{m,k,r''}(x) = \bigoplus_{i \in A_\ell} \bigwedge_{j=1}^{k+1} \bigoplus_{t=1}^{r''} x_{i,j,t}, \qquad \text{where } r'' = \frac{qr'}{2} \gg 1.$$

Having $r'' \geq 1$ is crucial for us because, together with $|A_\ell| \geq m/2$, it means that $\text{RW}_{m,k,r''}$ contains a perfect copy of $\text{GIP}_{m/2,k+1}$ (i.e. by possibly restricting and renaming some variables of $\text{RW}_{m,k,r''}$ and possibly negating the result, we obtain a function identical to $\text{GIP}_{m/2,k+1}$).

▶ Remark. We refine the definition of $\mathcal{R}$ to require that in (c) it use an $L(\ell)$ as specified above at each leaf ℓ.

Summarizing, the above discussion establishes that $\text{RW}_{m,k,r}$ "retains structure" with high probability under a random $\boldsymbol{\rho} \leftarrow \mathcal{R}$. The formal statement of this result (incorporating also Lemma 8) is as follows:

▶ **Lemma 13.** *Let F be any $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0{}_d$ circuit of size $s = n^{\tau \log n}$. The fair distribution $\mathcal{R}$ over restrictions $\boldsymbol{\rho} \in \{0,1,*\}^n$ from Lemma 8 satisfies the following: With probability $1 - \gamma_{\text{SL}} - \gamma_{\text{target}}$ over a draw of $\boldsymbol{\rho} \leftarrow \mathcal{R}$, both of the following hold:*

- **(i)** *$F \restriction \boldsymbol{\rho}$ belongs to $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AND}_k$;*
- **(ii)** *$\text{RW}_{m,k,r} \restriction \boldsymbol{\rho}$ contains a perfect copy of $\text{GIP}_{m/2,k+1}$.*

4 Bounding the correlation between the approximator and target post-restriction

With Lemma 13 in hand it is a simple matter to finish the argument. Fix any outcome ρ of $\boldsymbol{\rho} \leftarrow \mathcal{R}$ such that $F \restriction \rho$ and $\text{RW}_{m,k,r} \restriction \rho$ satisfy (i) and (ii) of Lemma 13. Applying either Fact 3 or Theorem 15 (depending on whether the top gate of F is SYM or THR) along with the lower bound of Theorem 14, we get that

$$\Pr_{\boldsymbol{x} \leftarrow \{0,1\}^n} \left[(F \restriction \rho)(\boldsymbol{x}) = (\text{RW}_{m,k,r} \restriction \rho)(\boldsymbol{x}) \right] \leq \frac{1}{2} + \exp\left(-\Omega(m/4^k)\right) = 1/2 + \gamma_{\text{corr}}, \tag{5}$$

where

$$\gamma_{\text{corr}} = \exp\left(-\Omega(m/4^k)\right) = \exp\left(-\Omega(m^{0.999})\right) = \exp\left(-\Omega(n^{0.499})\right).$$

This gives ingredient (3) as described in Section 1.4. Recalling the discussion at the end of Section 1.4, Theorem 3 follows from Lemma 13 and (5).

References

1. Miklós Ajtai. Σ^1_1-formulae on finite structures. *Annals of Pure and Applied Logic*, 24(1):1–48, 1983.
2. Miklós Ajtai and Avi Wigderson. Deterministic simulation of probabilistic constant depth circuits. In *Proc. 26th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 11–19, 1985.
3. László Babai, Noam Nisan, and Márió Szegedy. Multiparty protocols, pseudorandom generators for logspace, and time-space trade-offs. *J. Comput. System Sci.*, 45(2):204–232, 1992. 21st Symposium on the Theory of Computing (STOC). doi:10.1016/0022-0000(92)90047-M.

4 Richard Beigel and Jun Tarui. On ACC. *Computational Complexity*, 4:350–366, 1994.

5 Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudorandom bits. *SIAM J. Comput.*, 13(4):850–864, 1984. doi:10.1137/0213053.

6 Andrej Bogdanov. Pseudorandom generators for low degree polynomials. In *STOC'05: Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 21–30. ACM, New York, 2005. doi:10.1145/1060590.1060594.

7 Andrej Bogdanov and Emanuele Viola. Pseudorandom bits for polynomials. *SIAM J. Comput.*, 39(6):2464–2486, 2010. doi:10.1137/070712109.

8 Nader Bshouty. On learning multivariate polynomials under the uniform distribution. *Information Processing Letters*, 61(3):303–309, 1997.

9 Nader Bshouty and Yishay Mansour. Simple Learning Algorithms for Decision Trees and Multivariate Polynomials. *SIAM J. Comput.*, 31(6):1909–1925, 2002.

10 Shiteng Chen and Periklis A. Papakonstantinou. Depth reduction for composites. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science—FOCS 2016*, page to appear. IEEE, 2016.

11 Anindya De and Rocco Servedio. Efficient deterministic approximate counting for low-degree polynomial threshold functions. In *Proc. 46th Annual ACM Symposium on Theory of Computing (STOC)*, pages 832–841, 2014.

12 Ilias Diakonikolas, Daniel M. Kane, and Jelani Nelson. Bounded independence fools degree-2 threshold functions. In *Proc. 51st IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 11–20, 2010.

13 Ilias Diakonikolas, Homin Lee, Kevin Matulef, Krzysztof Onak, Ronitt Rubinfeld, Rocco Servedio, and Andrew Wan. Testing for concise representations. In *Proc. 48th Ann. Symposium on Computer Science (FOCS)*, pages 549–558, 2007.

14 Ilias Diakonikolas, Homin Lee, Krzysztof Matulef, Rocco Servedio, and Andrew Wan. Efficiently testing sparse GF(2) polynomials. *Algorithmica*, July 2010.

15 Ilias Diakonikolas, Ryan O'Donnell, Rocco Servedio, and Yi Wu. Hardness results for agnostically learning low-degree polynomial threshold functions. In *SODA*, pages 1590–1606, 2011.

16 A. Ehrenfeucht and M. Karpinski. The computational complexity of (xor,and)-counting problems. Technical report, preprint, 1989.

17 M. Goldmann. On the power of a threshold gate at the top. *Information Processing Letters*, 63(6):287–293, 1997.

18 D. Grigoriev, M. Karpinski, and M. Singer. Fast parallel algorithms for sparse multivariate polynomial interpolation over finite fields. *SIAM Journal on Computing*, 19(6):1059–1063, 1990.

19 Kristoffer Arnsfelt Hansen and Peter Bro Miltersen. Some meet-in-the-middle circuit lower bounds. In *Mathematical foundations of computer science 2004*, volume 3153 of *Lecture Notes in Comput. Sci.*, pages 334–345. Springer, Berlin, 2004. doi:10.1007/978-3-540-28629-5_24.

20 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.

21 Johan Håstad. On the correlation of parity and small-depth circuits. *SIAM Journal on Computing*, 43(5):1699–1708, 2014.

22 Johan Håstad and Mikael Goldmann. On the power of small-depth threshold circuits. *Comput. Complexity*, 1(2):113–129, 1991. doi:10.1007/BF01272517.

23 Russell Impagliazzo, William Matthews, and Ramamohan Paturi. A satisfiability algorithm for AC^0. In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 961–972, 2012.

24 A. Kalai, A. Klivans, Y. Mansour, and R. Servedio. Agnostically learning halfspaces. *SIAM Journal on Computing*, 37(6):1777–1805, 2008.

25 Daniel Kane. A Structure Theorem for Poorly Anticoncentrated Gaussian Chaoses and Applications to the Study of Polynomial Threshold Functions. In *FOCS*, pages 91–100, 2012.

26 M. Karpinski. Boolean circuit complexity of algebraic interpolation problems. "ICSI technical report", 1989.

27 M. Karpinski and M. Luby. Approximating the Number of Zeros of a $GF[2]$ Polynomial. *Journal of Algorithms*, 14:280–287, 1993.

28 M. Krause and P. Pudlak. Computing boolean functions by polynomials and threshold circuits. *Computational Complexity*, 7(4):346–370, 1998.

29 Shachar Lovett. Unconditional pseudorandom generators for low-degree polynomials. *Theory Comput.*, 5:69–82, 2009. doi:10.4086/toc.2009.v005a003.

30 Shachar Lovett and Srikanth Srinivasan. Correlation bounds for poly-size AC^0 circuits with $n^{1-o(1)}$ symmetric gates. In *Approximation, randomization, and combinatorial optimization*, volume 6845 of *Lecture Notes in Comput. Sci.*, pages 640–651. Springer, Heidelberg, 2011. doi:10.1007/978-3-642-22935-0_54.

31 Michael Luby, Boban Veličković, and Avi Wigderson. Deterministic approximate counting of depth-2 circuits. In *Proceedings of the 2nd ISTCS*, pages 18–24, 1993.

32 Raghu Meka and David Zuckerman. Pseudorandom generators for polynomial threshold functions. In *STOC*, pages 427–436, 2010.

33 M. Minsky and S. Papert. *Perceptrons: an introduction to computational geometry*. MIT Press, Cambridge, MA, 1968.

34 Cody Murray and Ryan Williams. Circuit lower bounds for nondeterministic quasi-polytime: An easy witness lemma for NP and NQP. In *Proceedings of the 50th Annual Symposium on Theory of Computing (STOC)*, 2018.

35 Noam Nisan. Pseudorandom bits for constant depth circuits. *Combinatorica*, 11(1):63–70, 1991.

36 Noam Nisan. The communication complexity of threshold gates. In *Combinatorics, Paul Erdős is eighty, Vol. 1*, Bolyai Soc. Math. Stud., pages 301–315. János Bolyai Math. Soc., Budapest, 1993.

37 Noam Nisan and Avi Wigderson. Hardness vs. randomness. *J. Comput. System Sci.*, 49(2):149–167, 1994. doi:10.1016/S0022-0000(05)80043-1.

38 V. V. Podolskii. Perceptrons of large weight. *Problems of Information Transmission*, 45(1):46–53, 2009.

39 Alexander Razborov and Avi Wigderson. $n^{\Omega(\log n)}$ lower bounds on the size of depth-3 threshold circuits with AND gates at the bottom. *Inform. Process. Lett.*, 45(6):303–307, 1993. doi:10.1016/0020-0190(93)90041-7.

40 R. Roth and G. Benedek. Interpolation and approximation of sparse multivariate polynomials over $GF(2)$. *SIAM J. Comput.*, 20(2):291–314, 1991.

41 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, August 22-26, 2016 - Kraków, Poland*, pages 82:1–82:16, 2016.

42 R. Schapire and L. Sellie. Learning sparse multivariate polynomials over a field with queries and counterexamples. *J. Comput. & Syst. Sci.*, 52(2):201–213, 1996.

43 Rocco A. Servedio and Li-Yang Tan. Improved pseudorandom generators from pseudorandom multi-switching lemmas, 2018. Manuscript. Available at https://arxiv.org/abs/1801.03590.

44 Adi Shamir. On the generation of cryptographically strong pseudorandom sequences. In *Automata, languages and programming (Akko, 1981)*, volume 115 of *Lecture Notes in Comput. Sci.*, pages 544–550. Springer, Berlin-New York, 1981.

45 Luca Trevisan and Tongke Xue. A derandomized switching lemma and an improved derandomization of AC^0 . In *Proceedings of the 28th IEEE Conference on Computational Complexity*, pages 242–247, 2013.

46 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1-3):1–336, 2012.

47 Emanuele Viola. Pseudorandom bits for constant-depth circuits with few arbitrary symmetric gates. *SIAM J. Comput.*, 36(5):1387–1403 (electronic), 2006/07. doi:10.1137/050640941.

48 Emanuele Viola. *On the power of small-depth computation*. Now Publishers Inc, 2009.

49 Emanuele Viola. The sum of d small-bias generators fools polynomials of degree d. *Comput. Complexity*, 18(2):209–217, 2009. doi:10.1007/s00037-009-0273-5.

50 Ryan Williams. Non-uniform ACC Circuit Lower Bounds. In *CCC 2011*, pages 115–125, 2011.

51 Andrew Yao. Theory and applications of trapdoor functions. In *23rd Annual Symposium on Foundations of Computer Science (Chicago, Ill., 1982)*, pages 80–91. IEEE, New York, 1982.

52 Andrew Yao. On ACC and threshold circuits. In *Proceedings of the Thirty-First Annual Symposium on Foundations of Computer Science*, pages 619–627, 1990.

A Preliminaries

We use bold font like $\boldsymbol{x}$, $\boldsymbol{\rho}$, etc. to denote random variables.

We write "size-S $\mathsf{AC}^0{}_d$" to denote the class of circuits of depth d consisting of at most S unbounded fan-in AND/OR gates with variables and negated variables as the inputs (we include these literals in the gate count).

Pseudorandomness. For $r < n$, we say that a distribution $\mathcal{D}$ over $\{0,1\}^n$ can be *sampled efficiently with r random bits* if (i) $\mathcal{D}$ is the uniform distribution over a multiset of size exactly 2^r of strings from $\{0,1\}^n$, and (ii) there is a deterministic algorithm $\mathrm{Gen}_{\mathcal{D}}$ which, given as input a uniform random r-bit string $\boldsymbol{x} \leftarrow \{0,1\}^r$, runs in time $\mathrm{poly}(n)$ and outputs a string drawn from $\mathcal{D}$.

For $\delta > 0$ and a class $\mathcal{C}$ of functions from $\{0,1\}^n$ to $\{0,1\}$, we say that a distribution $\mathcal{D}$ over $\{0,1\}^n$ *δ-fools $\mathcal{C}$ with seed length r* if (a) $\mathcal{D}$ can be sampled efficiently with r random bits via algorithm $\mathrm{Gen}_{\mathcal{D}}$, and (b) for every function $f \in \mathcal{C}$, we have

$$\left| \mathop{\mathbf{E}}_{\boldsymbol{s} \leftarrow \{0,1\}^r} [f(\mathrm{Gen}_{\mathcal{D}}(\boldsymbol{s}))] - \mathop{\mathbf{E}}_{\boldsymbol{x} \leftarrow \{0,1\}^n} [f(\boldsymbol{x})] \right| \leq \delta.$$

Equivalently, we say that $\mathrm{Gen}_{\mathcal{D}}$ is a *δ-PRG for $\mathcal{C}$ with seed length r*.

Restrictions. A *restriction* ρ of variables $x_1, \ldots, x_n$ is an element of $\{0,1,*\}^n$). Given a function $f(x_1, \ldots, x_n)$ and a restriction ρ, we write $f \restriction \rho$ to denote the function obtained by fixing x_i to $\rho(i)$ if $\rho(i) \in \{0,1\}$ and leaving x_i unset if $\rho(i) = *$. For two restrictions $\rho, \rho' \in \{0,1,*\}^n$, their *composition*, denoted $\rho\rho' \in \{0,1,*\}^n$, is the restriction defined by

$$(\rho\rho')_i = \begin{cases} \rho_i & \text{if } \rho_i \in \{0,1\} \\ \rho'_i & \text{otherwise.} \end{cases}$$

We write $\mathcal{R}_p$ to denote the standard distribution over random restrictions with $*$-probability p, i.e. $\boldsymbol{\rho}$ drawn from $\mathcal{R}_p$ is a random string in $\{0,1,*\}$ obtained by independently setting each coordinate to $*$ with probability p and to each of $0,1$ with probability $\frac{1-p}{2}$.

A.1 Multiparty communication complexity

We recall a celebrated lower bound of Babai, Nisan, and Szegedy [3] on the multi-party "number on forehead" (NOF) communication complexity of the generalized inner product function:

▶ **Theorem 14** ([3]). *There is a partition of the $m \cdot (k+1)$ inputs of*

$$\mathrm{GIP}_{m,k+1}(x) := \bigoplus_{i=1}^{m} \bigwedge_{j=1}^{k+1} x_{i,j}$$

into $k+1$ blocks such that the following holds: Let P be a $(k+1)$-party randomized NOF communication protocol exchanging at most $\frac{1}{10}(m/4^{k+1} - \log(1/\gamma_{\mathrm{comm}}))$ bits of communication and computing a Boolean function f with error γ_{err} (meaning that on every input x the protocol outputs the correct value $f(x)$ with probability at least $1-\gamma_{\mathrm{err}}$). Then

$$\Pr_{\boldsymbol{x} \leftarrow \{0,1\}^{m(k+1)}} \left[f(\boldsymbol{x}) = \mathrm{GIP}_{m,k+1}(\boldsymbol{x}) \right] \leq \frac{1}{2} + \gamma_{\mathrm{err}} + \gamma_{\mathrm{comm}}.$$

The connection between $\mathsf{SYM} \circ \mathsf{AND}_k$ circuits and $(k+1)$-party communication complexity is due to the following simple but influential observation of Håstad and Goldmann:

▶ **Fact 3** ([22]). *Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function computed by a size-s $\mathsf{SYM} \circ \mathsf{AND}_k$ circuit. Then for any partition of the n inputs of f into $k+1$ blocks, there is a deterministic NOF $(k+1)$-party communication protocol that computes f using $O(k \log s)$ bits of communication.*

For $\mathsf{THR} \circ \mathsf{AC}^0$ circuits we use an analogous result from [36] on the $(k+1)$-party randomized γ-error communication complexity of $\mathsf{THR} \circ \mathsf{AND}_k$ circuits:

▶ **Theorem 15** ([36]). *Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function computed by a $\mathsf{THR} \circ \mathsf{AND}_k$ circuit. Then for any partition of the n inputs of f into $k+1$ blocks, there is a randomized NOF $(k+1)$-party communication protocol that computes f with error γ_{err} using $O(k^3 \log n \log(n/\gamma_{\mathrm{err}}))$ bits of communication.*

B Applying the [37] paradigm to obtain pseudorandom generators from correlation bounds

A function f is said to be (s,τ)-*hard for a circuit class* $\mathcal{C}$ if every circuit $C \in \mathcal{C}$ of size at most s has $\Pr_{\boldsymbol{x}}[f(\boldsymbol{x}) = C(\boldsymbol{x})] \leq \frac{1}{2} + \tau$, where $\boldsymbol{x}$ is a uniform random input string. If this holds then we say that f gives a *correlation bound* of τ against $\mathcal{C}$-circuits of size s.

Given a quadruple (m,r,ℓ,s) of non-negative integers, a family $\mathcal{F} = \{T_1, \ldots, T_s\}$ of r-element subsets of $[m]$ is said to be an (m,r,ℓ,s)-*design* if for any two distinct subsets $T_i, T_j \in \mathcal{F}$ we have $|T_i \cap T_j| \leq \ell$.

An ANY_t gate is a gate that takes in t inputs and computes an arbitrary function from $\{0,1\}^t$ to $\{0,1\}$.

We recall the Nisan-Wigderson [37] translation from correlation bounds to PRGs:

▶ **Theorem 16** (The Nisan-Wigderson generator). *Fix a circuit class $\mathcal{C}$ and let $m, r, \ell, s \in \mathbb{N}$ be positive parameters with $m \geq r \geq \ell$. Given an explicit $f : \{0,1\}^r \to \{0,1\}$ that is $(s \cdot 2^\ell, \varepsilon/s)$-hard for $\mathcal{C} \circ \mathsf{ANY}_{\log \ell}$ and an explicit (m, r, ℓ, s)-design, there is an explicit PRG $G : \{0,1\}^m \to \{0,1\}^s$ that ε-fools size-s circuits in $\mathcal{C}$. (Hence for $s \geq n$, by taking the first n output bits of G there is an explicit PRG mapping $\{0,1\}^m$ to $\{0,1\}^n$ that ε-fools size-s n-variable circuits in $\mathcal{C}$.)*

The existence of explicit designs is well known, in particular we recall the following:

▶ **Lemma 17** (Problem 3.2 of [46]). *There is a deterministic algorithm which, for any $r, s \in \mathbb{N}$, runs in time $\mathrm{poly}(m, s)$ and outputs an explicit (m, r, ℓ, s)-design with $m = O(r^2/s)$ and $\ell \leq \log s$.*

A PRG from the [47] correlation bound. Viola [47] gives an explicit function $f : \{0,1\}^r \to \{0,1\}$ and shows that for every constant d there is a constant c_d such that f is $(r^{c_d \log r}, r^{-c_d \log r})$-hard for $\mathsf{SYM} \circ \mathsf{AC}^0{}_d$. Fix any d. Given values for ε, s let us set the parameters

$$\ell = \log s, \qquad r = 2^{10 \cdot \sqrt{\frac{1}{c_d} \log(s/\varepsilon)}}.$$

It is straightforward to verify that $s \cdot 2^\ell \leq r^{c_d \log r}$ and $\varepsilon/s \geq r^{-c_d \log r}$. By Lemma 17 there is an explicit (m, r, ℓ, s)-design with $m = O(r^2/\ell) = 2^{O(\sqrt{\frac{1}{c_d} \log(s/\varepsilon)})}$, so applying the Nisan-Wigderson generator, we get that for $s \geq n$, there is an explicit PRG $G : \{0,1\}^m \to \{0,1\}^n$ with seed length $m = 2^{O(\sqrt{\log(s/\varepsilon)})}$ that ε-fools n-variable size-s circuits in $\mathsf{AC}^0{}_d$.

A PRG from the [30] correlation bound. Lovett and Srinivasan [30] give an explicit function $f : \{0,1\}^r \to \{0,1\}$ such that for every constant d there is a constant c_d such that f is $(r^{c_d \log \log r}, \exp(-r^{1-o(1)})$-hard for $\mathsf{SYM} \circ \mathsf{AC}^0{}_d$. We proceed as above but now choosing

$$\ell = \log s, \qquad r = 2^{\frac{10}{c_d} \cdot \frac{\log s}{\log \log s}} + (\log(s/\varepsilon))^{1+o(1)}.$$

It is straightforward to verify that $s \cdot 2^\ell \leq r^{c_d \log \log r}$ and $\varepsilon/s \geq \exp(-r^{1-o(1)})$. By Lemma 17 there is an explicit (m, r, ℓ, s)-design with $m = O(r^2/\ell) = 2^{O(\log s/ \log \log s)} \cdot (\log(1/\varepsilon))^{2+o(1)}$, so applying the Nisan-Wigderson generator, we get that for $s \geq n$, there is an explicit PRG $G : \{0,1\}^m \to \{0,1\}^n$ with seed length $m = 2^{O(\log s/ \log \log s)} + (\log(1/\varepsilon))^{2+o(1)}$ that ε-fools n-variable size-s circuits in $\mathsf{SYM} \circ \mathsf{AC}^0{}_d$.

For $\mathsf{THR} \circ \mathsf{AC}^0{}_d$, [30] show that the same function f is $(r^{c_d \log \log r}, \exp(-r^{1/2-o(1)}))$-hard for $\mathsf{THR} \circ \mathsf{AC}^0{}_d$; a similar analysis to the above gives an explicit PRG $G : \{0,1\}^m \to \{0,1\}^n$ with seed length $m = 2^{O(\log s/ \log \log s)} + (\log(1/\varepsilon))^{4+o(1)}$ that ε-fools n-variable size-s circuits in $\mathsf{THR} \circ \mathsf{AC}^0{}_d$.

A PRG from our Theorem 3: Proof of Theorem 2. Theorem 3 gives an explicit $f : \{0,1\}^r \to \{0,1\}$ and $\tau > 0$ such that for all d, f is $(r^{\tau \log r}, \exp(-r^{0.499}))$-hard for $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0{}_d$. This time we choose

$$\ell = \log s, \qquad r = 2^{10 \cdot \sqrt{\frac{2}{\tau} \log s}} + (\log(s/\varepsilon))^{2.005}.$$

We have $s \cdot 2^\ell \leq r^{\tau \log r}$ and $\varepsilon/s \geq \exp(-r^{0.499})$, so we get that for $s \geq n$, there is an explicit PRG $G : \{0,1\}^m \to \{0,1\}^n$ with seed length $m = O(r^2/\ell) = 2^{O(\sqrt{\log s})} + (\log(1/\varepsilon))^{4.01}$ that ε-fools n-variable size-s circuits in $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0{}_d$.

A PRG from our Theorem 4: Proof of Corollary 5. Finally, Theorem 4 gives an explicit $f : \{0,1\}^r \to \{0,1\}$ and $\tau > 0$ such that for all d, f is $(r^{\tau \log r}, \exp(-r^{0.499}))$-hard for the class of depth-d circuits over $\{0,1\}^r$ that contain $r^{0.249}$ many SYM or THR gates. We choose ℓ, r as above, so similar to the above, we get that there is an explicit PRG $G : \{0,1\}^m \to \{0,1\}^n$ with seed length $m = O(r^2/\ell) = 2^{O(\sqrt{\log s})} + (\log(1/\varepsilon))^{4.01}$ that ε-fools n-variable size-s depth-d circuits with at most $2^{c\sqrt{\log s}}$ many SYM or THR gates.

C Proof of Theorem 4: Handling multiple SYM and THR gates

We prove Theorem 4 via a slight variant of Theorem 3 and an argument from [30] (a related argument appears in a somewhat different form in [47]). The variant of Theorem 3, stated as Theorem 20 below, is proved by combining ingredients (1), (2) and (3) as in Section 1.4, but now with the aim of proving a correlation bound against $\mathsf{ANY}_u \circ \{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0{}_d$ circuits rather than $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0{}_d$ circuits (where here and throughout this appendix we take $u := n^{0.249}$). As we describe at the end of this section, once this correlation bound against $\mathsf{ANY}_u \circ \{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0{}_d$ is in place, the extension to circuits with $n^{0.249}$ many SYM or THR gates directly follows using an argument from [30].

In more detail we have: (throughout the following the values of m, k, r are as they were before)

▶ **Lemma 18** (Lemma 8 analogue). *Fix $u := n^{0.249}$ and let F be an $\mathsf{ANY}_u \circ \{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0{}_d$ circuit where each of the u $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0{}_d$ subcircuits of F has size at most $s = n^{\tau \log n}$. There is a fair distribution $\mathcal{R}$ over restrictions $\boldsymbol{\rho} \in \{0,1,*\}^n$ such that the following holds: With probability $1 - \gamma_{\mathrm{SL}} = 1 - \exp(-\Omega_d(\sqrt{n/\log n}))$ over the draw of $\boldsymbol{\rho} \leftarrow \mathcal{R}$, it is the case that $F \restriction \boldsymbol{\rho}$ belongs to the class $\mathcal{F}_{simple,\,u} := \mathsf{ANY}_u \circ \{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AND}_k$.*

The proof is almost identical to that of Lemma 8, with $\mathsf{ANY}_u \cdot \{\mathsf{SYM}, \mathsf{THR}\}$ taking the place of $\{\mathsf{SYM}, \mathsf{THR}\}$ throughout the argument. Now in Corollary 10 the gate G corresponds to $\mathsf{ANY}_u \circ \{\mathsf{SYM}, \mathsf{THR}\}$ (rather than to just $\{\mathsf{SYM}, \mathsf{THR}\}$ as earlier) and the total circuit size of F is us rather than s (leading to $us \cdot 2^{-t}$ rather than $s \cdot 2^{-t}$ on the RHS of the Corollary 10 bound), but this is swallowed up by the slack in the inequalities leading to (1).

▶ **Lemma 19** (Lemma 13 analogue). *Fix $u := n^{0.249}$ and let F be an $\mathsf{ANY}_u \circ \{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0{}_d$ circuit where each $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0{}_d$ subcircuit has size at most $s = n^{\tau \log n}$. The fair distribution $\mathcal{R}$ over restrictions $\boldsymbol{\rho} \in \{0,1,*\}^n$ from Lemma 18 satisfies the following: With probability $1 - \gamma_{\mathrm{SL}} - \gamma_{\mathrm{target}}$ over a draw of $\boldsymbol{\rho} \leftarrow \mathcal{R}$, both of the following hold:*

- **(i)** *$F \restriction \boldsymbol{\rho}$ belongs to $\mathcal{F}_{simple,\,u} = \mathsf{ANY}_u \circ \{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AND}_k$; and*
- **(ii)** *$\mathrm{RW}_{m,k,r} \restriction \boldsymbol{\rho}$ contains a perfect copy of $\mathrm{GIP}_{m/2,k+1}$.*

The proof of Lemma 19 is unchanged from Section 3.

▶ **Theorem 20** (Theorem 3 analogue). *Fix $u := n^{0.249}$. There is an absolute constant $\tau > 0$ and an explicit $\mathrm{poly}(n)$-time computable function $H : \{0,1\}^n \to \{0,1\}$ with the following property: for any constant d, for n sufficiently large, for F an $\mathsf{ANY}_u \circ \{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0{}_d$ circuit where each $\{\mathsf{SYM}, \mathsf{THR}\} \circ \mathsf{AC}^0{}_d$ subcircuit has size at most $s = n^{\tau \log n}$, we have*

$$\Pr_{\boldsymbol{x} \leftarrow \{0,1\}^n}[F(\boldsymbol{x}) = \mathrm{RW}_{m,k,r}(\boldsymbol{x})] \leq \frac{1}{2} + \exp(-\Omega(n^{0.249})).$$

The proof, using Lemmas 18 and 19, is virtually identical to the proof of Theorem 3 using Lemmas 8 and 13. The only difference is that we use the obvious extensions of Fact 3 and Theorem 15 to $\mathsf{ANY}_u \cdot \mathsf{SYM} \circ \mathsf{AND}_k$ circuits and $\mathsf{ANY}_u \cdot \mathsf{THR} \circ \mathsf{AND}_k$ circuits respectively; these extensions are stated for completeness below.

▶ **Fact 4** (Fact 3 analogue). *Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function computed by a size-s $\mathsf{ANY}_u \circ \mathsf{SYM} \circ \mathsf{AND}_k$ circuit. Then for any partition of the n inputs of f into $k+1$ blocks, there is a deterministic NOF $(k+1)$-party communication protocol that computes f using $u \cdot O(k \log s)$ bits of communication.*

▶ **Theorem 21** (Theorem 15 analogue). *Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function computed by a $\mathsf{ANY}_u \circ \mathsf{THR} \circ \mathsf{AND}_k$ circuit. Then for any partition of the n inputs of f into $k+1$ blocks, there is a randomized NOF $(k+1)$-party communication protocol that computes f with error γ_{err} using $u \cdot O(k^3 \log n \log(n/\gamma_{\mathrm{err}}))$ bits of communication.*

We note that the extra factors of u in Fact 4 and in Theorem 21, which are not present in the analogous Fact 3 and Theorem 15 respectively, are responsible for the quantitatively weaker correlation bound in Theorem 4 as opposed to Theorem 3. The $n^{0.249}$ bound on the number of SYM or THR gates which Theorem 4 can handle can be traded off against the correlation bound in that theorem; we leave the details to the interested reader.

Finally, the correlation bound Theorem 4 follows from Theorem 20 exactly as Theorem 6 of [30] follows from Lemma 3 of that paper.

Randomly Coloring Graphs of Logarithmically Bounded Pathwidth

Shai Vardi[1]
Krannert School of Management, Purdue University, West Lafayette, IN, 47907, USA
svardi@purdue.edu

Abstract

We consider the problem of sampling a proper k-coloring of a graph of maximal degree Δ uniformly at random. We describe a new Markov chain for sampling colorings, and show that it mixes rapidly on graphs of logarithmically bounded pathwidth if $k \geq (1+\epsilon)\Delta$, for any $\epsilon > 0$, using a *hybrid paths* argument.

2012 ACM Subject Classification Theory of computation → Random walks and Markov chains, Mathematics of computing → Markov-chain Monte Carlo methods

Keywords and phrases Random coloring, Glauber dynamics, Markov-chain Monte Carlo

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.57

Related Version A full version of the paper is available at `https://hal.archives-ouvertes.fr/hal-01832102`.

Acknowledgements We thank Leonard Schulman for valuable discussions, and Adam Wierman and the anonymous reviewers for their useful comments.

1 Introduction

A (proper) k-coloring of a graph $G = (V, E)$ is an assignment $\sigma : V \to \{1, \ldots, k\}$ such that neighboring vertices have different colors. We consider the problem of sampling (almost) uniformly at random from the space of all k-colorings of a graph.[2] The problem has received considerable attention from the computer science community in recent years, e.g., [12, 17, 24, 26, 34, 45, 46]. It also has applications in Combinatorics (e.g., [5]) and Statistical Physics (e.g., [43]).

Sampling colorings (as well as other combinatorial objects, e.g., [6, 27, 38]) is commonly done using Markov Chain Monte Carlo (MCMC) methods. A large body of work on sampling colorings is devoted to analyzing a particular Markov chain, known as *Glauber dynamics*: Choose a vertex v uniformly at random; choose a color c uniformly at random from the set of available colors (the complement of the set of colors of the neighbors of v); recolor v with c. Jerrum [26] showed that the Glauber dynamics mix in time $O(n \log n)$ when $k > 2\Delta$, where Δ is the maximal degree of the graph. Vigoda [46] improved the bound on the number of colors to $k > 11\Delta/6$ using a different Markov chain, and showed that it mixes in time $O(nk \log n)$. This remains the best known bound on k for general graphs. A major open question is for what values of k can we sample colors efficiently (i.e., in polynomial time)? It is conjectured (e.g., [16]) that $k = \Delta + 2$ colors suffice, and furthermore, that the Glauber dynamics mix rapidly for any $k \geq \Delta + 2$.

[1] Supported in part by the Linde Foundation and NSF grants CNS-1254169 and CNS-1518941.
[2] We define precisely what we mean by "almost" in Section 2.2.

Table 1 Comparison of results on sampling k-colorings using MCMC.

Degree	Girth	Graph family*	$k >$	Dynamics	Mixing time	Ref.
any	any	any	2Δ	Glauber	$O(n \log n)$	[26]
any	any	any	$(1.833\ldots)\Delta$	Flip	$O(n \log n)$	[46]
$\Omega(\log n)$	$\Omega(\log \log n)$	any	$(1.763\ldots)\Delta$	Glauber	$O(n \log n)$	[11]
$\Omega(\log n)$	≥ 9	any	$(1+\epsilon)\Delta$	Glauber	$O(n \log n)$	[24]
$\geq \Delta_0$ †	≥ 5	any	$(1.763\ldots)\Delta$	Glauber	$O(n \log n)$	[13]
$\geq \Delta_0$ †	≥ 6	any	$(1.489\ldots)\Delta$	Glauber	$O(n \log n)$	[13]
$O(1)$	∞	Trees	4	Glauber	poly(n)	[35]
any	any	tw $= O(1)$	$\Delta + O\left(\sqrt{\Delta}\right)$	Glauber	poly(n)	[21]
$O(1)$	any	cw $= O(\log n)$	$\Delta + 2$	Glauber	poly(n)	[2]
any	any	pw $= O(\log n)$	$(1+\epsilon)\Delta$	Single-Flaw	poly(n)	Here

† Δ_0 is some absolute constant. * tw, cw and pw are short for treewidth, cutwidth and pathwidth resepctively.

A lot of work has focused on improving the bounds of Vigoda on restricted families of graphs. Dyer and Frieze [11] showed that if the maximal degree and girth are $\Omega(\log n)$, the Glauber dynamics mix in $O(n \log n)$ for $k > \alpha\Delta$, where $\alpha \approx 1.763$. The degree and girth requirements and the value of α were improved in a line of works [13,20,24,25,37]; see Table 1 for a comparison and summary of some milestones. The current state of the art results exhibit a tradeoff between the value of α and the degree and girth requirements. Hayes and Vigoda [24] showed that on graphs with $\Delta = \Omega(\log n)$ and girth at least 9, $(1+\epsilon)\Delta$ colors suffice to ensure fast mixing. On the other hand, the Glauber dynamics have been shown to mix rapidly on graphs with girth at least 5 (resp. 6) and $\Delta > \Delta_0$ (where Δ_0 is some absolute constant) using roughly 1.763Δ (resp. 1.489Δ) colors [13]. Stronger bounds have only been shown on highly specialized families of graphs, such as trees [36], planar graphs [22] and Erdös-Rényi graphs [12].

The two results that are closest to ours are the following: Hayes [21] showed that for graphs with maximum eigenvalue ρ, the mixing time of the Glauber dynamics is $O(n \log n)$, when $k > \Delta + c\rho$, where c is slightly greater than 1. Planar graphs, trees and graphs of bounded treewidth have a maximum eigenvalue of $O(\sqrt{\Delta})$, implying fast mixing for these graphs. Berger et al. [2] showed that for graph with bounded degree and logarithmically bounded cutwidth, the Glauber dynamics mix in polynomial time. We formally define pathwidth later on, but we note that the cutwidth of any graph G is at least as large as its pathwidth, which is at least as large as its treewidth; in addition, the pathwidth is at most $O(\log n)$ times the treewidth [30].

1.1 Results and Techniques

Our main result is an algorithm that efficiently samples a $((1+\epsilon)\Delta)$-coloring (almost) uniformly at random if the input graph has logarithmically bounded pathwidth, for any $\epsilon > 0$.[3]

▶ **Theorem 1** (Informal). *Let $\epsilon > 0$ and G be a graph with n vertices and maximal degree Δ. There exists an algorithm for sampling a $((1+\epsilon)\Delta)$-proper coloring of G (almost) uniformly*

[3] Assuming $(1+\epsilon)\Delta \geq \Delta + 1$.

at random, whose running time is polynomial in n, Δ, *and* $\epsilon^{-O(\mathrm{pw}(G))}$, *where* $\mathrm{pw}(G)$ *is the pathwidth of* G. *In particular, when* $\mathrm{pw}(G) = O(\log n)$, *the algorithm runs in polynomial time.*

The two main methods of bounding the mixing time of random walks are *coupling* and bounding the *spectral gap* of the transition matrix [19]; methods of bounding the conductance or congestion (which are used to bound the spectral gap) are generally considered to be stronger than coupling methods [19,31]. Despite this, most of the work on sampling colorings uses various coupling methods [10–13,20,22–26,37,46]; the Glauber dynamics do not lend themselves easily to the techniques that are usually used for bounding the spectral gap. In particular, bounding the *congestion* of the underlying graph of the transition matrix[4] typically involves defining flows between states in the underlying graph. If one can describe a flow in the underlying graph such that the congestion of each edge is not too large, it implies that the spectral gap is large and the Markov chain mixes rapidly [9,44]. It is not clear how to construct such flows for the Glauber dynamics, as a transition involves changing a vertex's color to one of its available colors: if all of the neighbors of a vertex v that is colored `blue` are colored `red`, how do we go about changing v's color to `red`?

We introduce a new Markov chain, which we call *Single-Flaw dynamics*. The difference between the Single-Flaw and Glauber dynamics is that the Single-Flaw dynamics also allow colorings that have a "single flaw" – there is at least one monochromatic edge, and all monochromatic edges share a vertex. In other words, the coloring is not proper, but there is a single vertex v such that we can reach a proper coloring by changing v's color only. We call such colorings *singly-flawed*. Concretely, the Single-Flaw dynamics Markov chain is the following: choose a vertex v and a color c at random. If changing v's color to c results in a coloring that is either proper or singly-flawed, change v's color to c. Otherwise do not.

The main advantage afforded by this Markov chain is that it allows us to define paths between two states (colorings) α and β: select some order on the vertices, $v_1, \ldots, v_n$. Starting from v_1, for each vertex v_i, change its color to $\beta(v_i)$. If the transition leads to a proper coloring, continue to v_{i+1}. Otherwise, "fix" the monochromatic edges by recoloring the neighbors of v_i that are also colored $\beta(v_i)$ (as $k \geq \Delta + 1$ there is always at least one available color). When there are no monochromatic edges remaining, continue to vertex v_{i+1}. We note that it is possible to first "fix" the neighbors of v and only then change the color of v, and this corresponds to the Glauber dynamics; we only lose a single color choice per neighbor (in the Single-Flaw dynamics we can recolor with the original color of v, whereas in the Glauber dynamics we cannot). Therefore, our analysis can be easily modified to apply to the Glauber dynamics. We chose to analyze the Single-Flaw dynamics for two reasons (in addition to using one fewer color): 1) The analysis is cleaner this way, and 2) It is straightforward to extend the analysis techniques to "Multi-Flaw" dynamics, which we believe could prove useful in other sampling and counting settings as well.

We first describe a simple attempt to adapt the canonical paths argument of Jerrum and Sinclair [27,28] to our setting, using the canonical paths described above. Although it fails in all non-trivial cases, it is instructive as it exemplifies an important part of our method. For every edge in the underlying graph, (try to) describe an injective function from paths going through the edge to the state space of the chain. If we can describe such a function, it would mean that at most $|\Omega|$ paths use each edge. If there were no "fixing phase" (i.e., the graph was disconnected), this would be easy: Let t be the transition from the state σ such

[4] In the underlying graph of the transition matrix A of a Markov chain, the states are represented by vertices, there is an edge (u, v) between two states $u, v \in \Omega$ with weight $w_e = A_{u,v}$ if $A_{u,v} \neq 0$.

that the color of the j^{th} vertex is changed to c. The injective function would map the path from α to β to the following coloring σ^*: for vertices $v_i : i = 1, 2, \dots j$, $\sigma^* = \alpha(v_i)$, for all other vertices $v_i : i = j+1, \dots, n$, $\sigma^*(v_i) = \beta(v_i)$. The mapping is injective because (i) σ^* a proper coloring and (ii) knowing t and σ^* allows us to recover α and β, as

$$\sigma(v_i) = \begin{cases} \beta(v_i) & i = 1, 2, \dots, j \\ \alpha(v_i) & \text{otherwise.} \end{cases}$$

This would imply that at most $|\Omega|$ paths would use each transition. We note that it is not necessary to show that *at most* $|\Omega|$ paths use each transition to show polynomial time mixing; it suffices to show that $|\Omega| \cdot \text{poly}(n)$ paths use each transition [27]. If we are guaranteed that at every round, at most r vertices will be colored differently from α and β, this means that at most

$$(k-1)^r |\Omega| \tag{1}$$

paths use each transition. This observation was used by Berger et al. [2], who used the cutwidth to bound r. (The cutwidth of a graph G is the smallest integer r such that there exists a labeling $v_1, \dots, v_n$ of the vertices such that for all $1 \leq k \leq n$, the number of edges from $\{v_1, \dots, v_k\}$ to $\{v_{k+1}, \dots, v_n\}$ is at most r.) We use a similar argument, but are able to leverage the pathwidth of the graph, which gives a stronger result, as the cutwidth of any graph is at least as large as its pathwidth. In essence, a small pathwidth implies that there is some order on the vertices, such that if our canonical paths obey this order, not too many vertices will need to be fixed at any time. We note that finding such an order is NP-hard [4], however we only need that such an order exists for the canonical paths argument.

The main contribution in this paper is removing the requirement that the maximal degree is bounded by a constant; in other words, removing the number of colors from the base in Expression (1). In order to do this, we use a multicommodity flow argument [9, 44]. Instead of specifying a single path for each pair of states, we describe a flow between them in the underlying graph. Whenever we fix a vertex's color, we split the flow evenly among all available options. This is similar to the argument of Morris and Sinclair [38], in which flow is also split up among different paths; it helps route the flow more evenly, thereby avoiding the case where one edge is heavily congested while other "available" edges are not. In contrast to [38], we only split the flow in the fixing stages; in fact, whenever a vertex v_i is colored $\beta(v_i)$, this consolidates the flow! Our technique can be thought of as a hybrid argument between the canonical paths proof technique of [27] and the multicommodity flow argument of [38]. We call these paths *hybrid paths* and believe they could be useful in other scenarios as well.

Finally, recall that the state space of the Single-Flaw dynamics includes flawed colorings. We show that there are not too many singly-flawed colorings relative to proper colorings, hence executing the chain polynomially many times will guarantee that we output a proper coloring w.h.p.

1.2 Paper organization

The proofs of Corollary 3, Corollary 4, Lemma 6, Lemma 10, Lemma 11, Lemma 14, Lemma 15 and Theorem 16 can be found in the appendix.

1.3 Related work

Most of the work on sampling colorings has focused on the Glauber dynamics (e.g., [12,13,17, 23,26,34,35,37,39,45]). Other Markov chains have been analyzed, notably the Flip dynamics of Vigoda [46], which is closely related to the chain proposed by Wang, Swendsen, and Kotecký [47]: when a vertex v is required to change color from c to c', the colors of the entire neighborhood that is colored with c and c' are flipped (the chain of Vigoda only performs some flips with some probability). It is also possible to sample colorings using approaches that do not use MCMC methods; for example, Efthymiou [14] proposed a combinatorial method for sampling colors that does not use a Markov chain, and used it to show that it is possible to sample colorings on $G(n, d/n)$ using $k > (1+\epsilon)d$ colors. A caveat is that the run time is only polynomial w.p. $1 - 2n^{-2/3}$. The main difference between Single-Flaw dynamics and other work on sampling colors is that we allow improper states. There are other Markov chains that also consist of "flawed" states that are not part of the space we wish to sample from. An example is the Markov chain for sampling perfect matchings in bipartite graphs, proposed by Broder [7] and analyzed by Jerrum and Sinclair [27] and Jerrum, Sinclair and Vigoda [28]: the chain consists of perfect matchings (of size n), and imperfect matchings of size $n-1$. An interesting distinction is that Broder's chain needs the imperfect matchings to transition between perfect matchings (otherwise, it is unclear how to transition). We do not need the imperfect states to show convergence: the Glauber dynamics are known to converge to the uniform distribution for $k \geq \Delta + 2$; we only use the imperfect colorings to bound the mixing time. We do get the additional benefit that the Single-Flaw dynamics are guaranteed to converge for $k \geq \Delta + 1$ (in contrast to the Glauber dynamics, which are not—a simple counterexample is a triangle graph); we do not leverage this in this work.

The method of bounding the conductance or congestion of the transition matrix of a Markov chain has been used to great success in sampling and counting of various problems e.g., [18, 28, 38]. To our knowledge, the only places that these types of arguments have been successfully applied to sampling colorings is in bounding the mixing time of the Glauber dynamics on trees with bounded degree [35] and hyperbolic graphs [2].

Sampling colorings corresponds to sampling configurations of the zero temperature k-state anti-ferromagnetic Potts model [40]. One can draw an analogy between our technique and temperature-tuned walks that also include higher energy levels (see e.g., [33]), though instead of walking at a fixed temperature, which would allow some Poisson-like distribution of the number of flaws, we allow exactly one flaw, and correct it before allowing the next flaw.

The terms *treewidth* and *pathwidth* were introduced by Robertson and Seymour [41, 42]; the concept of treewidth was discovered independently several times, and was originally introduced under a different name by Bertelè and Brioschi [3]. Of particular interest is a work by Chekuri, Khanna and Shepherd [8] that consider multicommodity flows on graphs of bounded treewidth. Their techniques and results are incomparable to ours; they study the flow on graphs, while we use a flow to bound the congestion on the underlying graph of the Single-Flaw dynamics.

2 Preliminaries

We denote the set $\{1, 2, \ldots, m\}$ by $[m]$. Let $G = (V, E)$ be a graph, and denote $|V| = n$. We assume that the vertices of G are uniquely identified by $\{1, 2, \ldots, n\}$. For any (not necessarily simple) path p in G, let $|p|$ denote the length of p (i.e., the number of edges in p, where if an edge appears k times in p, it is counted k times).

2.1 Colorings

For any k-coloring of G, $\sigma : V \to [k]$, let $MCE(G) = \{(u, v) : \sigma(u) = \sigma(v)\}$ denote the set of monochromatic edges. σ is a proper coloring if $MCE(G) = \emptyset$. σ is a *singly-flawed coloring* if $MCE(G) \neq \emptyset$ and there is a vertex that is common to all edges in $MCE(G)$, i.e., $\exists v : \forall e \in MCE(G), v \in e$. We say that such a vertex v is a *flawed vertex* of σ. Note that a singly-flawed coloring has exactly two flawed vertices if $|MCE(G)| = 1$ and one flawed vertex otherwise. We denote the set of proper k-colorings of G colors by $\mathcal{C}_p(G, k)$ and the set of all singly-flawed colorings by $\mathcal{C}_{sf}(G, k)$. We drop G and k when they are clear from context. Let σ be a coloring. If, after recoloring some $v \in V$ with a color c, there is no monochromatic edge (u, v), we say that c is *available* to v in σ. Note that a color's availability does not depend on whether σ or the coloring obtained by recoloring v with c is proper, singly-flawed or otherwise.

We first show two results that will be useful later on, regarding proper and singly-flawed colorings: (1) the ratio of singly-flawed colorings to proper colorings is "not too large" (Corollary 3), and (2) there is a mapping from singly-flawed colorings to proper colorings, such that "not too many" singly-flawed colorings are mapped to any proper coloring (Corollary 4). Both results are corollaries of the following simple lemma.

▶ **Lemma 2.** *For any $G = (V, E)$ such that $|V| = n$ and $k \geq \Delta + 2$, there exists a surjective function*

$$g : \mathcal{C}_p(G, k) \times [k] \times [n] \to \mathcal{C}_{sf}(G, k).$$

Proof. For every coloring $\sigma \in \mathcal{C}_p(G, k)$, every vertex $v \in V$ and every color $c \in [k]$, let

$$\sigma'_{c,v} = \begin{cases} \sigma(u) & \text{if } u \neq v \\ c & \text{if } u = v \end{cases}$$

If $\sigma'_{c,v} \in \mathcal{C}_{sf}(G, k)$, let $g(\sigma, c, v) = \sigma'_{c,v}$, otherwise let $g(\sigma, c, v)$ be some arbitrary coloring in $\mathcal{C}_{sf}(G, k)$. It is easy to see that every $\sigma' \in \mathcal{C}_{sf}(G, k)$ is in the range of g: the reverse operation of changing the color of a flawed vertex v in σ' to some available color gives a proper coloring. ◀

The two corollaries that we require are the following.

▶ **Corollary 3.** *For any $G = (V, E)$ such that $|V| = n$ and $k \geq \Delta + 2$, $|\mathcal{C}_{sf}(G, k)| \leq kn|\mathcal{C}_p(G, k)|$.*

▶ **Corollary 4.** *For any $G = (V, E)$ such that $|V| = n$ and $k \geq \Delta + 2$, there exists a function $g' : \mathcal{C}_{sf}(G, k) \to \mathcal{C}_p(G, k)$, for which each element in the co-domain has at most kn pre-images in the domain.*

2.2 Markov chains and rapid mixing

In this section we review some of the results on the mixing time of Markov chains that we will require. The reader is referred to [32] for an excellent introduction to Markov chains and modern techniques on bounding their mixing time.

Consider a discrete-time Markov chain $\mathcal{MC}$ with finite state space Ω and symmetric transition probability matrix P (i.e., $P(\sigma, \sigma') = P(\sigma', \sigma)$ for all $\sigma, \sigma' \in \Omega$). The chain is said to be *irreducible* if for every pair of states $\sigma, \sigma' \in \Omega$, there exists some t such that $P^t(\sigma, \sigma') > 0$; in other words, it is possible to get from any state to any state using a finite

number of transitions. It is *aperiodic* if for any $\sigma, \in \Omega$, $\gcd\{t : P^t(\sigma, \sigma) > 0\} = 1$. It is *lazy* if for all $\sigma \in \Omega, P(\sigma, \sigma) \geq 1/2$. A fundamental theorem of stochastic processes states that an irreducible and aperiodic Markov chain converges to a unique *stationary distribution* π over Ω, i.e., $\lim_{t\to\infty} P^t(\sigma, \sigma') = \pi(\sigma')$ for all $\sigma, \sigma' \in \Omega$. If in addition P is symmetric, then π is uniform over Ω (e.g., [1]).

Our goal is to describe a *fully-polynomial almost uniform sampler for proper colorings*; namely, a randomized algorithm that, given as inputs a graph $G = (V, E)$ and a bias parameter δ, outputs a random proper coloring of G from a distribution D that satisfies $d_{TV}(D, U) \leq \delta$, where U is the uniform distribution on the proper colorings of G and d_{TV} is the total variation distance, defined as follows:[5] For any two distributions μ, ν on Ω,

$$d_{TV}(\mu, \nu) = \max_{S\subseteq\Omega} |\mu(S) - \nu(S)|. \tag{2}$$

We are interested in the rate at which a Markov chain converges to its stationary distribution π. We define the *mixing time* from a state σ to be

$$\tau_\sigma(\delta) = \min\{\bar{t} : d_{TV}(P^t(\sigma, \cdot), \pi) \leq \delta \text{ for all } t \geq \bar{t}\}, \tag{3}$$

We further define the mixing time of the Markov chain to be $\tau(\delta) = \max_\sigma \tau_\sigma(\delta)$. We say that a Markov chain is *rapidly mixing* if $\tau(1/2e)$ is polynomial in n. The constant $1/2e$ is arbitrary, as a bound on $\tau(1/2e)$ implies a bound on $\tau(\delta)$ for any $\delta > 0$ (e.g., [1]):

$$\tau(\delta) \leq (1 - \log \delta) \cdot \tau(1/2e).$$

In order to bound the mixing time, we describe a multicommodity flow on the underlying graph $H = (\Omega, F)$ of the Markov chain, where $F = \{(\sigma, \sigma'), P(\sigma, \sigma') > 0\}$ is the set of all transitions that have positive probability.

We denote by

$$q(\sigma, \sigma') = \pi(\sigma)P(\sigma, \sigma'), \tag{4}$$

the *ergodic flow* through the edge (σ, σ') of H (an intuitive way to think about $q(\sigma, \sigma')$ is the probability of traversing edge (σ, σ') at stationarity.)

For all ordered pairs $(\alpha, \beta) \in \Omega^2$, let $\mathcal{P}_{\alpha,\beta}$ denote a set of (not necessarily simple) directed paths from α to β in H. A *flow* is a function $f : \mathcal{P} \to \mathcal{R}^+ \cup \{0\}$ where $\mathcal{P} = \bigcup_{\alpha,\beta} \mathcal{P}_{\alpha,\beta}$ that satisfies

$$f_{\alpha,\beta} \equiv \sum_{p\in\mathcal{P}_{\alpha,\beta}} f(p) = \pi(\alpha)\pi(\beta), \tag{5}$$

for every $\alpha, \beta \in \Omega$.

We define the congestion on an edge (σ, σ') with respect to a flow f by

$$\rho_f(\sigma, \sigma') = \frac{1}{q(\sigma, \sigma')} \sum_{\alpha,\beta\in\Omega} \sum_{p:(\sigma,\sigma')\in p\in\mathcal{P}_{\alpha,\beta}} f(p)|p|, \tag{6}$$

and the congestion of f by

$$\rho_f = \max_{(\sigma,\sigma')\in F} \rho_f(\sigma, \sigma').$$

[5] Alternatively, we can define it as $d_{TV}(\mu, \nu) = \frac{1}{2}\sum_{\sigma\in\Omega} |\mu(\sigma) - \nu(\sigma)|$. It is easy to verify that the two definitions are equivalent.

We use the following theorem, due to Sinclair [44] and Diaconis and Stroock [9], that relates the mixing time to the congestion of a flow. Note that it holds for any flow; in order to bound the mixing time, we need to find *some* flow that has low congestion.

▶ **Theorem 5** ([44]). *For any irreducible, aperiodic, lazy and symmetric Markov chain $\mathcal{MC}$ with transition matrix P on state space Ω, any flow f on the underlying graph of $\mathcal{MC}$, and any state $\sigma_0 \in \Omega$,*

$$\tau_{\sigma_0}(\delta) \leq \rho_f \left(\ln \pi(\sigma_0)^{-1} + \ln \delta^{-1}\right).$$

2.3 Pathwidth and vertex separation

A *path decomposition* of a graph $G = (V, E)$ is a path P with m nodes, where each node of P represents a subset of V: $X_i \subseteq V, i \in [m]$ such that the following hold:

1. $\bigcup_{i=1}^{m} X_i = V$,
2. For every $(u, v) \in E$, $u, v \in X_i$ for some $i \in [m]$.
3. For all $i, j, k \in [m]$, if X_j is on the (unique) path between X_i and X_k, then $X_i \cap X_k \subseteq X_j$.

The first requirement guarantees that every vertex of G is in at least one node of P, the second that every two neighboring vertices in G share at least one node in P, and the third that if some vertex $v \in V$ is in both X_i and X_k, it is in every node of the path between X_i and X_k in P. The width of P is defined as $\max_{i \in [m]}\{|X_i| - 1\}$. The *pathwidth* of G, denoted $\mathrm{pw}(G)$, is the minimal ω such that there exists some path-decomposition of G with width ω. The *treewidth* is similarly defined, when P is a tree.

A linear ordering of a graph $G = (V, E)$ is a bijective mapping of vertices to integers; $L : V \to \{1, 2, \ldots, n\}$. Given a graph G, a linear ordering L, and an integer $j \in \{1, \ldots, n\}$, let A_j be the set of vertices mapped to the integers $1, \ldots, j$ by L; i.e., $A_j = \{v : L(v) \leq j\}$. Let B_j denote the set of vertices that are mapped to integers greater than j by L: $B_j = V \setminus A_j$. A *minimal vertex separator* for an index $j \in \{1, \ldots n\}$ (denoted $\mathrm{MVS}(G, L, j)$) is a minimal set of vertices $S_j \subset V$ such that the following hold:[6]

1. For all $u \in S_j$, $L(u) > j$.
2. A_j and $B_j \setminus S_j$ are disconnected; that is, there is no edge $(u, v) \in E$ such that $u \in A_j, v \in B_j \setminus S_j$.

W.l.o.g., we henceforth assume for simplicity that L is the identity ordering; i.e., $\forall j \in \{1, \ldots, n\}$, $L(j) = j$.

The *vertex separator number* for a graph G, denoted $\mathrm{VSN}(G)$ is the minimal size of the largest minimal vertex separator over all possible linear orderings L of G.

$$\mathrm{VSN}(G) = \min_{L} \max_{j \in \{1,\ldots,n\}} \{|\,\mathrm{MVS}(G, L, j)|\}.$$

We call an order L for which $\mathrm{VSN}(G)$ is minimized a *minimal order* for G.

We require the following lemma and theorem.

▶ **Lemma 6.** *Let G and L be a graph and a linear order thereof. For $j = 1, \ldots, n$, set $S_j = \mathrm{MVS}(G, L, j)$. For all $j \in \{2, \ldots, n\}$, $S_{j-1} \subseteq S_j \cup \{j\}$.*

The proof of Lemma 6 can be found in the appendix.

▶ **Theorem 7** ([29]). *For any graph G, $\mathrm{VSN}(G) = \mathrm{pw}(G)$.*

[6] Traditionally, a vertex separator for j is defined such that the separating subset appears before j in the ordering [15]. This is identical to our definition with the order inverted.

3 Single-Flaw dynamics

Let $G = (V, E)$ be a graph with maximal degree Δ and $\epsilon > 0$ such that $(1+\epsilon)\Delta \geq \Delta + 2$. The state space Ω of Markov chain $\mathcal{MC}(G, \epsilon)$ (or simply $\mathcal{MC}$) is the set of all proper and singly-flawed k-colorings of G, for $k = \lceil (1+\epsilon)\Delta \rceil$: $\Omega = \mathcal{C}_p(G, k) \cup \mathcal{C}_{sf}(G, k)$. For simplicity, we henceforth assume that $\epsilon\Delta$, $(1+\epsilon)\Delta$ and $(1+\epsilon)\epsilon^{-1}$ are integers. It is easy to generalize the results to real values thereof. For $\sigma \in \Omega$, the transitions $\sigma \to \sigma'$ of $\mathcal{MC}$ are the following

- Let $\sigma' = \sigma$.
- With probability 1/2, do nothing (laziness).
- Otherwise, choose a vertex v and color c uniformly at random from V and $[k]$ respectively. Tentatively, set $\sigma'(v) = c$.
- If $\sigma' \notin \Omega$, set $\sigma'(v) = \sigma(v)$.

It is easy to verify that the chain is irreducible, aperiodic, lazy and symmetric; hence the conditions of Theorem 5 hold, and it remains to describe a flow with low congestion of the underlying graph of $\mathcal{MC}$. Our main result describes such a flow.

▶ **Lemma 8.** *Let $\epsilon > 0$ and G be a graph with maximal degree Δ. Then there exists a flow f on the underlying graph of the Markov chain $\mathcal{MC}(G, \epsilon)$ such that*

$$\rho_f \leq 8\,\mathrm{pw}(G)(1+\epsilon)^3\Delta^3 n^5 \left((1+\epsilon)\epsilon^{-1}\right)^{2\,\mathrm{pw}(G)}.$$

There are at most k^n possible colorings of G. Because the stationary distribution is uniform, for all $\sigma \in \Omega$, $\pi(\sigma) \geq \frac{1}{k^n}$, hence $\ln \pi(\sigma)^{-1} = O(n \log n)$.

Theorem 5 and Lemma 8 imply the following theorem, which is our main result.

▶ **Theorem 9.** *Let $\epsilon > 0$ and G be a graph with maximal degree Δ. The mixing time of $\mathcal{MC}(G, \epsilon)$ satisfies*

$$\tau(\delta) = O\left(\mathrm{pw}(G)(1+\epsilon)^3\Delta^3 n^5 \left((1+\epsilon)\epsilon^{-1}\right)^{2\,\mathrm{pw}(G)} \left(n \log n + \ln \delta^{-1}\right)\right).$$

In particular, if $\mathrm{pw}(G) = O(\log n)$, the chain mixes in polynomial time.

In order to prove Lemma 8, we design a flow for $\mathcal{MC}$. To do so, we first describe the set of paths that we will route the flow through.

3.1 Hybrid paths

Let L be some minimal order of G. For simplicity and w.l.o.g. we assume that L is the identity order, i.e., $L(i) = i$ for all $i \in [n]$. We remark that we do not need to explicitly find L; we only require its existence for the hybrid paths argument. Let $\lambda = \frac{\mathrm{VSN}(G)}{\log n}$. If $\mathrm{pw}(G) = O(\log n)$, as is assumed here, λ is a constant.

For each $j \in [n]$, let $S_j = \mathrm{MVS}(G, L, j)$ be a minimal vertex separator. A hybrid path is an ordered set of colorings. We denote the set of hybrid paths from α to β by $\gamma_{\alpha,\beta}$. *For the rest of this subsection and the next, we assume that α and β are both proper* colorings; we will extend the sets of paths to include ones that start and/or end at singly-flawed colorings in Section 3.3. Each hybrid path consists of n phases, where phase j consists of $|S_j| + 1$ steps, for a total of $\ell = n + \sum_{j=1}^{n} |S_j| \leq (\lambda + 1)(n \log n)$ steps. Each step is a recoloring of some vertex; it is possible that a vertex is "recolored" with the same color. In that case, the state (coloring) does not change, but we still count this redundant recoloring as a step, as it guarantees that all paths are of the same length; this will help to make the analysis more

concise. A state that appears at the start of the ℓ^{th} step of the j^{th} phase of a hybrid path in $\gamma_{\alpha,\beta}$ is said to be at *distance* (j,ℓ) from α; alternatively, we say that it happens at *time* (j,ℓ).[7] We denote the states of $\gamma_{\alpha,\beta}$ that are at distance (j,ℓ) from α by $\Phi_{\alpha,\beta}(j,\ell)$. The set of hybrid paths $\gamma_{\alpha,\beta}$ can be thought of as a layered graph, where all states of $\Phi_{\alpha,\beta}(j,\ell)$ are placed in the same layer.

3.1.1 A phase of the hybrid paths

We describe a single phase of $\gamma_{\alpha,\beta}$. For any $j \in [n]$, all states in $\Phi_{\alpha,\beta}(j,1)$ are proper colorings. Note that $\Phi_{\alpha,\beta}(1,1) = \{\alpha\}$. For **the first step** of the j^{th} phase, for every $\sigma_i \in \Phi_{\alpha,\beta}(j,1)$, set

$$\sigma'_i(v) = \begin{cases} \sigma_i(v) & \text{if } v \neq j \\ \beta(j) & \text{if } v = j \end{cases}.$$

We therefore have that $\Phi_{\alpha,\beta}(j,2) = \bigcup_i \sigma'_i$. It is clear that there is only one way to route the flow entering σ_i: it is all routed to σ'_i on (σ_i, σ'_i). It is possible that flow becomes consolidated in this step: the flow from all states $\sigma_i \in \Phi_{\alpha,\beta}(j,1)$ that differ only in the j^{th} coordinate is routed to the same σ'_i. Note that every $\sigma' \in \Phi_{\alpha,\beta}(j,2)$ is either a proper or a singly-flawed coloring.

The ℓ^{th} step in the j^{th} phase, $\ell \in \{2,3,\ldots|S_j|+1\}$ is a *splitting step*, and is the following: Let u_ℓ be the (lexicographically) $(\ell-1)^{th}$ vertex of S_j. For every $\sigma_i \in \Phi_{\alpha,\beta}(j,\ell)$, let $C_i(u_\ell)$ be the set of colors available to u_ℓ under σ_i. For each $\sigma_i \in \Phi_{\alpha,\beta}(j,\ell)$ and color $c \in C_i(u_\ell)$, let

$$\sigma'_{i,c}(v) = \begin{cases} \sigma_i(v) & \text{if } v \neq u_\ell \\ c & \text{if } v = u_\ell \end{cases}.$$

We have that $\Phi_{\alpha,\beta}(j,\ell+1) = \bigcup_i \sigma'_{i,c}$. From each state $\sigma_i \in \Phi_{\alpha,\beta}(j,\ell)$, the flow is split evenly among the transitions (i.e., a $1/|C_i(u_\ell)|$ fraction of the flow entering σ_i is routed on each $(\sigma_i, \sigma'_{i,c})$). Note that all states in $\Phi_{\alpha,\beta}(j,|S_j|+2)$ are proper colorings, as any edge that may have been monochromatic in any $\sigma \in \Phi_{\alpha,\beta}(j,2)$ will have been recolored. For all $1 \leq j < n$, set $\Phi_{\alpha,\beta}(j+1,1) = \Phi_{\alpha,\beta}(j,|S_j|+2)$. Note that $S_n = \emptyset$ and $\Phi_{\alpha,\beta}(n,2) = \{\beta\}$.

Because for all j, $|S_j| = O(\log n)$, given α, β, j and ℓ, the color of most vertices in $\Phi_{\alpha,\beta}(j,\ell)$ is uniquely determined. In particular, for $\ell > 1$, denote $A_j = \{v : v \leq j\}$ and $B_j = V \setminus (A_j \cup S_j)$.[8] It must hold that for any $\sigma \in \Phi_{\alpha,\beta}(j,\ell)$, $v_a \in A_j$, and $v_b \in B_j$, $\sigma(v_a) = \beta(v_a)$ and $\sigma(v_b) = \alpha(v_b)$. This is because all the vertices in A_j have been recolored to β and will not be recolored again, while the vertices in B_j have no neighbors in A_j, hence they have not been recolored yet.

We denote by $QS(\alpha,\beta,j,\ell)$ (QS stands for "quantum set") the set of vertices whose color is not uniquely defined by α, β, j, ℓ. By Lemma 6, it holds that $QS(\alpha,\beta,j,\ell) \subseteq S_j$, but equality does not necessarily hold: assume that $S_{j-1} \subset S_j$, and let $u \neq j$ be some vertex in $S_j \setminus S_{j-1}$. Then u's color is still $\alpha(u)$ at time $(j,2)$, as it has not yet been recolored, even though $u \in S_j$.

We note that $QS(\alpha,\beta,j,\ell)$ does not in fact depend on α or β. In fact,

---

[7] Note that a state can be at several distances if it appears more than once on the path.

[8] For $\ell = 1$, we consider $(j-1, |S_{j-1}|+2)$ instead of $(j,1)$, unless $j = 1$, in which case the vertices are all colored by α.

▶ **Lemma 10.** *$QS(\alpha, \beta, j, \ell)$ is uniquely determined by either*
1. *j and ℓ, or*
2. *j and (σ, σ').*

The proof of Lemma 10 can be found in the appendix.

Due to Lemma 10, we sometimes refer to $QS(\alpha, \beta, j, \ell)$ by $QS(j, (\sigma, \sigma'))$ or $QS(j, \ell)$.

3.2 Bounding the flow

Let α and β be proper colorings, $t = (\sigma, \sigma') \in F$ be some transition, and $j \in [n]$ be an integer. Denote the flow routed through t from α to β in phase j by $f_{j,t,\alpha,\beta}$. Note that we are only considering the flow routed through t **during phase** j; it is possible that the hybrid path passes through t in several phases, possibly carrying a different flow each time. Intuitively, it seems natural that after the flow was split evenly several times, "not too much" flow is routed through any state, as it has a specific combination of colors of the vertices of $QS(j, t)$, and many such combinations are possible. It is not straightforward to show this, however, as the colors of different vertices in each state are not independent. The difficulty is compounded by the fact that flow is consolidated at the first step of every phase. Nevertheless, we can prove the following lemma, whose proof can be found in the full version of the paper, by rearranging the vertices of $QS(j, t)$ and using an inductive reasoning on this new order.

▶ **Lemma 11.** *The flow routed from α to β through any $t = (\sigma, \sigma') \in F$ in any phase $j \in [n]$ is at most*

$$f_{j,t,\alpha,\beta} \leq \frac{\pi(\alpha)\pi(\beta)}{(\epsilon\Delta)^{|QS(j,t)|}}.$$

The proof of Lemma 11 can be found in the appendix.

We want to bound the total flow through a transition in any single phase. The following set of recoloring functions χ is useful. Let C be a set of (available) colors. χ_C is a function, parameterized by C, that takes as an input a color $c \in [k]$. Its output is a color from C, such that each color in C has the same number of pre-images, up to one. We do not explicitly define χ, only note that such a set of functions exists. For example, if $k = 13$, $C = \{1, 2, 3, 4, 5\}$, χ_C could allocate $(c \mod 5) + 1$ to every $c \in [k]$, giving each color in C either two or three pre-images. We make the following observation (recall we assume $\epsilon\Delta$, $(1+\epsilon)\Delta$ and $(1+\epsilon)\epsilon^{-1}$ are integers).

▶ **Lemma 12.** *For χ_C as defined above, if $k = (1+\epsilon)\Delta$ and $|C| \geq \epsilon\Delta$, each color in C has at most $(1+\epsilon)\epsilon^{-1}$ pre-images in $[k]$.*

Armed with Lemma 11 and the functions χ, we are now ready to bound the total flow through a transition in any single phase.

▶ **Lemma 13.** *The flow $f_{j,t}$ routed through any $t = (\sigma, \sigma') \in F$ in any phase $j \in [n]$ satisfies*

$$f_{j,t} \leq \pi(\cdot)^2 |\mathcal{C}_p| \cdot \left((1+\epsilon)\epsilon^{-1}\right)^{2|S_j|},$$

where $\pi(\cdot)$ is the probability of any state at stationarity.

Proof. For each (j, t), where $j \in [n]$ and $t = (\sigma, \sigma') \in F$, denote by $\texttt{pairs}_{j,t}$ the set of pairs of states $\alpha, \beta \in \mathcal{C}_p^2$ whose paths pass through t in phase j. We describe a function $\mu_{j,t}$ whose domain is $\texttt{pairs}_{j,t}$. We view the co-domain of $\mu_{j,t}$ as the Cartesian product of 3 sets X, Y and Z: $\mu_{j,t} : \texttt{pairs}_{j,t} \to X \times Y \times Z$; the output of $\mu_{j,t}(\cdot)$ is a triple (x, y, z). The function

will be injective, therefore the size of the co-domain of $\mu_{j,t}$ will serve as an upper bound to $|\texttt{pairs}_{j,t}|$.

The sets X, Y, Z are the following.

- X is the set of all proper colorings. Assume that the input to $\mu_{j,t}$ is some pair (α, β). In the coloring specified by x, all vertices in A_j[9] are colored by α. All vertices in B_j are colored by β. Before specifying the coloring of S_j under x, note that already, together with j and t, this allows us to deduce α completely on all vertices in $V \setminus QS(j,t)$ and β on all vertices $V \setminus S_j$. To determine the colors of S_j in x, we color them one at a time, using χ. This information, while not characterizing $\beta(v)$ completely for $v \in S_j$, allows allows us to restrict the possible value of $\beta(v)$ to a set of size at most $(1+\epsilon)\epsilon^{-1}$ possible values.
- Y is $\left[(1+\epsilon)\epsilon^{-1}\right]^{|S_j|}$, allowing us to pinpoint $\beta(v)$ for every $v \in S_j$.
- Finally, Z is simply all possible colorings of the vertices of $QS(j,t)$ under α.

Clearly x, y, z, j and t allow us to recover α and β. The size of the co-domain is at most

$$|\mathcal{C}_p| \cdot \left((1+\epsilon)\epsilon^{-1}\right)^{|S_j|} \cdot k^{|QS(j,t)|}.$$

Combining with Lemma 11 we get that the total flow through any transition t at phase j is at most

$$\begin{aligned} f_{j,t} &\leq |\mathcal{C}_p| \cdot \left((1+\epsilon)\epsilon^{-1}\right)^{|S_j|} \cdot k^{|QS(j,t)|} \cdot \frac{\pi(\cdot)\pi(\cdot)}{(\epsilon\Delta)^{|QS(j,t)|}} \\ &= \pi(\cdot)^2 |\mathcal{C}_p| \left((1+\epsilon)\epsilon^{-1}\right)^{|S_j|} \cdot \frac{((1+\epsilon)\Delta)^{|QS(j,t)|}}{(\epsilon\Delta)^{|QS(j,t)|}} \\ &\leq \pi(\cdot)^2 |\mathcal{C}_p| \cdot \left((1+\epsilon)\epsilon^{-1}\right)^{2|S_j|}, \end{aligned}$$

where the last inequality is because $QS(j,t) \subseteq S_j$ for any t. ◀

3.2.1 The congestion of an edge

We are ready to prove our main result of the section, that the congestion of any edge $(\sigma, \sigma') \in F$ under the flow defined by the hybrid paths from proper coloring to proper colorings, is polynomial in the number of vertices.

▶ **Lemma 14.** *The congestion of any transition t under f, when f is restricted to flows from proper colorings to proper colorings, satisfies*

$$\rho_f(t) \leq 2k(\lambda+1)n^3 \log n \left((1+\epsilon)\epsilon^{-1}\right)^{2\,\mathrm{pw}(G)}.$$

The proof of Lemma 14 can be found in the appendix.

3.3 Mixing time

Lemma 14 applies to the congestion from flow between proper colorings only. We extend this result to all of f. We rephrase our main lemma:

▶ **Lemma 15.** *The congestion of any transition t under f satisfies*

$$\rho_f(t) \leq 8k^3(\lambda+1)n^5 \log n \left((1+\epsilon)\epsilon^{-1}\right)^{2\,\mathrm{pw}(G)}.$$

The proof of Lemma 15 can be found in the appendix.

[9] As before, $A_j = \{v : v \leq j\}$, except for $\ell = 0$, for which $A_j = \{v : v < j\}$.

4 The sampling algorithm

In order to sample a proper coloring, we need to execute the Markov chain sufficiently many times to guarantee that w.h.p. it outputs a proper coloring, and when it does, return that coloring. The pseudo code is given as Algorithm 1 in Appendix A. For graphs of pathwidth bounded by $O(\log n)$, the algorithm runs time polynomial in n and $\log \delta$, where δ is the required bias parameter.

▶ **Theorem 16.** *Algorithm 1 is a fully polynomial almost uniform sampler for proper colorings with bias parameter δ.*

The proof of Theorem 16 can be found in the appendix.

References

1 David J. Aldous. Random walks on finite groups and rapidly mixing markov chains. *Séminaire de probabilités de Strasbourg*, 17:243–297, 1983. URL: `http://eudml.org/doc/113445`.

2 Noam Berger, Claire Kenyon, Elchanan Mossel, and Yuval Peres. Glauber dynamics on trees and hyperbolic graphs. *Probability Theory and Related Fields*, 131(3):311–340, 2005.

3 Umberto Bertelè and Francesco Brioschi. *Nonserial Dynamic Programming*. Academic Press, Inc., Orlando, FL, USA, 1972.

4 H.L. Bodlaender, J.R. Gilbert, H. Hafsteinsson, and T. Kloks. Approximating treewidth, pathwidth, frontsize, and shortest elimination tree. *Journal of Algorithms*, 18(2):238–255, 1995.

5 G.R. Brightwell and P. Winkler. Random coloring of a Cayley tree. *Contemporary combintorics*, 10:247–276, 2002.

6 Graham Brightwell and Peter Winkler. Counting linear extensions is #P-complete. In *Proceedings of the Twenty-third Annual ACM Symposium on Theory of Computing*, STOC '91, pages 175–181, 1991.

7 Andrei Z. Broder. How hard is to marry at random? (on the approximation of the permanent). In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing, STOC '86*, pages 50–58, 1986.

8 Chandra Chekuri, Sanjeev Khanna, and F. Bruce Shepherd. A note on multiflows and treewidth. *Algorithmica*, 54(3):400–412, 2009.

9 Persi Diaconis and Daniel Stroock. Geometric bounds for eigenvalues of markov chains. *Ann. Appl. Probab.*, 1(1):36–61, 02 1991. `doi:10.1214/aoap/1177005980`.

10 Martin E. Dyer, Abraham D. Flaxman, Alan M. Frieze, and Eric Vigoda. Randomly coloring sparse random graphs with fewer colors than the maximum degree. *Random Struct. Algorithms*, 29(4):450–465, 2006.

11 Martin E. Dyer and Alan M. Frieze. Randomly coloring graphs with lower bounds on girth and maximum degree. *Random Struct. Algorithms*, 23(2):167–179, 2003.

12 Martin E. Dyer and Alan M. Frieze. Randomly coloring random graphs. *Random Struct. Algorithms*, 36(3):251–272, 2010.

13 Martin E. Dyer, Alan M. Frieze, Thomas P. Hayes, and Eric Vigoda. Randomly coloring constant degree graphs. *Random Struct. Algorithms*, 43(2):181–200, 2013.

14 Charilaos Efthymiou. A simple algorithm for sampling colorings of g(n, d/n) up to the gibbs uniqueness threshold. *SIAM J. Comput.*, 45(6):2087–2116, 2016.

15 J. A. Ellis, I. H. Sudborough, and J. S. Turner. The vertex separation and search number of a graph. *Inf. Comput.*, 113(1):50–79, 1994.

16 Alan Frieze and Eric Vigoda. A survey on the use of markov chains to randomly sample colorings. In *Combinatorics, Complexity and Chance*. Oxford University Press, 2007. `doi:10.1093/acprof:oso/9780198571278.003.0004`.

17 Leslie Ann Goldberg, Mark Jerrum, and Marek Karpinski. The mixing time of Glauber dynamics for coloring regular trees. *Random Struct. Algorithms*, 36(4):464–476, 2010.

18 Heng Guo and Mark Jerrum. Random cluster dynamics for the ising model is rapidly mixing. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '17*, pages 1818–1827, 2017.

19 Venkatesan Guruswami. Rapidly mixing markov chains: A comparison of techniques (A survey). *CoRR*, abs/1603.01512, 2016. `arXiv:1603.01512`.

20 Thomas P. Hayes. Randomly coloring graphs of girth at least five. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing, June 9-11, 2003, San Diego, CA, USA*, pages 269–278, 2003.

21 Thomas P. Hayes. A simple condition implying rapid mixing of single-site dynamics on spin systems. In *47th Annual IEEE Symposium on Foundations of Computer Science (FOCS*, pages 39–46, 2006.

22 Thomas P. Hayes, Juan Carlos Vera, and Eric Vigoda. Randomly coloring planar graphs with fewer colors than the maximum degree. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, '07*, pages 450–458, 2007.

23 Thomas P. Hayes, Juan Carlos Vera, and Eric Vigoda. Randomly coloring planar graphs with fewer colors than the maximum degree. *Random Struct. Algorithms*, 47(4):731–759, 2015.

24 Thomas P. Hayes and Eric Vigoda. A non-markovian coupling for randomly sampling colorings. In *44th Symposium on Foundations of Computer Science (FOCS '03), Proceedings*, pages 618–627, 2003.

25 Thomas P. Hayes and Eric Vigoda. Coupling with the stationary distribution and improved sampling for colorings and independent sets. *Ann. Appl. Probab.*, 16(3):1297–1318, 2006.

26 Mark Jerrum. A very simple algorithm for estimating the number of k-colorings of a low-degree graph. *Random Struct. Algorithms*, 7(2):157–166, 1995.

27 Mark Jerrum and Alistair Sinclair. Approximating the permanent. *SIAM J. Comput.*, 18(6):1149–1178, 1989.

28 Mark Jerrum, Alistair Sinclair, and Eric Vigoda. A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries. *J. ACM*, 51(4):671–697, 2004.

29 Nancy G. Kinnersley. The vertex separation number of a graph equals its path-width. *Information Processing Letters*, 42(6):345–350, 1992.

30 Ephraim Korach and Nir Solel. Tree-width, path-width, and cutwidth. *Discrete Applied Mathematics*, 43(1):97–101, 1993.

31 V. S. Anil Kumar and H. Ramesh. Coupling vs. conductance for the jerrum-sinclair chain. *Random Struct. Algorithms*, 18(1):1–17, 2001.

32 David A. Levin, Yuval Peres, and Elizabeth L. Wilmer. *Markov chains and mixing times*. American Mathematical Society, 2006.

33 Jun S. Liu. *Monte Carlo Strategies in Scientific Computing*. Springer Publishing Company, Incorporated, 2008.

34 Pinyan Lu, Kuan Yang, Chihao Zhang, and Minshen Zhu. An FPTAS for counting proper four-colorings on cubic graphs. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017*, pages 1798–1817, 2017.

35 Brendan Lucier and Michael Molloy. The Glauber dynamics for colorings of bounded degree trees. *SIAM J. Discrete Math.*, 25(2):827–853, 2011.

36 Fabio Martinelli, Alistair Sinclair, and Dror Weitz. Fast mixing for independent sets, colorings and other models on trees. In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '04*, pages 456–465, 2004.

37 Michael Molloy. The Glauber dynamics on colorings of a graph with high girth and maximum degree. *SIAM J. Comput.*, 33(3):721–737, 2004.

38 Ben Morris and Alistair Sinclair. Random walks on truncated cubes and sampling 0-1 knapsack solutions. *SIAM J. Comput.*, 34(1):195–226, 2004.

39 Elchanan Mossel and Allan Sly. Gibbs rapidly samples colorings of g(n, d/n). *Probability Theory and Related Fields*, 148(1):37–69, Sep 2010.

40 R. B. Potts. Some generalized order-disorder transformations. *Mathematical Proceedings of the Cambridge Philosophical Society*, 48(1):106–109, 1952. doi:10.1017/S0305004100027419.

41 Neil Robertson and P.D. Seymour. Graph minors. i. excluding a forest. *Journal of Combinatorial Theory, Series B*, 35(1):39–61, 1983.

42 Neil Robertson and P.D Seymour. Graph minors. iii. planar tree-width. *Journal of Combinatorial Theory, Series B*, 36(1):49–64, 1984.

43 J. Salas and A.D. Sokal. Absence of phase transition for antiferromagnetic Potts models via the Dobrushin uniqueness theorem. *J Stat Phys*, 86:551–579, 1997.

44 Alistair Sinclair. Improved bounds for mixing rates of Markov chains and multicommodity flow. *Combinatorics, Probability & Computing*, 1:351–370, 1992.

45 Prasad Tetali, Juan Carlos Vera, Eric Vigoda, and Linji Yang. Phase transition for the mixing time of the Glauber dynamics for coloring regular trees. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '10*, pages 1646–1656, 2010.

46 Eric Vigoda. Improved bounds for sampling colorings. In *40th Annual Symposium on Foundations of Computer Science, FOCS '99*, pages 51–59, 1999.

47 Jian-Sheng Wang, Robert H. Swendsen, and Roman Kotecký. Antiferromagnetic potts models. *Phys. Rev. Lett.*, 63:109–112, Jul 1989.

A The sampling algorithm

Algorithm 1: An almost-uniform sampler for proper colorings.

Input : $G = (V, E)$ with maximal degree Δ, a number of colors $k \geq \Delta + 2$, a bias parameter $\delta > 0$

Output : a proper k-coloring of G

Set $\epsilon = \lceil \frac{k}{\Delta} \rceil$;
Set $\delta_1 = \delta/(kn+1)^2$;
Set $T = \lceil \ln(3/\delta)(kn+2)^2 \rceil$;
for $t = 1$ *to* T **do**
  Simulate $\mathcal{MC}(G, \epsilon)$ for $\tau(\delta_1)$ steps, starting from an arbitrary proper coloring;
  If the final state σ is a proper coloring, return σ;
Return an arbitrary proper coloring;

B Missing proofs

Proof of Lemma 2

Proof. For every coloring $\sigma \in \mathcal{C}_p(G,k)$, every vertex $v \in V$ and every color $c \in [k]$, let

$$\sigma'_{c,v} = \begin{cases} \sigma(u) & \text{if } u \neq v \\ c & \text{if } u = v \end{cases}$$

If $\sigma'_{c,v} \in \mathcal{C}_{sf}(G,k)$, let $g(\sigma,c,v) = \sigma'_{c,v}$, otherwise let $g(\sigma,c,v)$ be some arbitrary coloring in $\mathcal{C}_{sf}(G,k)$. It is easy to see that every $\sigma' \in \mathcal{C}_{sf}(G,k)$ is in the range of g: the reverse operation of changing the color of a flawed vertex v in σ' to some available color gives a proper coloring. ◀

Proof of Corollary 3

Proof. Immediate from the surjectivity of the function g in Lemma 2. ◀

Proof of Corollary 4

Proof. For every $\sigma' \in \mathcal{C}_{sf}$, arbitrarily select one pre-image (σ,v,c) w.r.t. g, and set σ as the image for σ' under g'. ◀

Proof of Lemma 6

Proof. For any $j \in \{1,\ldots,n\}$, S_j is uniquely determined: it is exactly the neighbors of the vertices of A_j in B_j. This is because for every $v \in A_j$, the set $N(v) \cap B_j$ must all be in S_j, otherwise there is an edge between v and B_j. From the minimality of S_j, therefore, $S_j = \bigcup_{v \in A_j} N(v) \cap B_j$. Hence,

- If $j \in S_{j-1}$ then $S_j = S_{j-1} \setminus \{j\} \cup (N(j) \cap B_j)$.
- If $j \notin S_{j-1}$ then $S_j = S_{j-1} \cup (N(j) \cap B_j)$.

This completes the proof of the lemma. ◀

Proof of Lemma 10

Proof. For any α,β, the same vertex is recolored at (j,ℓ): at $\ell = 1$, vertex j is recolored; in all other instances, u_ℓ is recolored by at least $\epsilon\Delta$ different colors, regardless of α,β. Further, note that once a vertex u is in such a "quantum state", it will remain in quantum state until it is colored $\beta(u)$ at distance $(u,1)$. Therefore, although we cannot recover the exact transitions used without knowledge of α,β, the set of vertices whose color is unknown at any given time is fixed. For the second observation, notice that (σ,σ') recolors some specific vertex u_ℓ (even if it is an idle recoloring), hence ℓ can be inferred. ◀

Proof of Lemma 11

Proof. From Lemma 10, (j,t) uniquely defines (j,ℓ). If no flow is routed from α to β through t in phase j, the lemma is trivially satisfied. Otherwise, we show that

$$f_{j,\ell,\alpha,\beta} \leq \frac{\pi(\alpha)\pi(\beta)}{(\epsilon\Delta)^{|QS(j,\ell)|}},$$

where $f_{j,\ell,\alpha,\beta}$ is the maximal flow from α to β through any t at distance (j,ℓ) from α. Order the vertices of $QS(j,\ell)$ in reverse order of the time since their last color change

(possibly a null color change). That is, the vertex whose color changed most recently is last in the order. Let $M = |QS(j,\ell)|$ and relabel the vertices of $QS(j,\ell)$ by $1,\ldots,M$ according to their place in this order. Similarly, relabel $\Phi_{\alpha,\beta}(j',\ell')$, by $\Phi_1,\ldots,\Phi_M$, where Φ_i is the set of states at the time just *after* vertex i last changed its color. In other words, $\Phi_M = \Phi_{\alpha,\beta}(j,\ell+1)$,$\Phi_{M-1} = \Phi_{\alpha,\beta}(j,\ell)$, and so on. It is possible that for some m, Φ_m corresponds to states in the previous phase, i.e., $\Phi_m = \Phi_{\alpha,\beta}(j-1,\ell')$. Note that we drop the α,β from the notation for clarity, but we are still only considering the flow from α to β.

We now show by that for any set of $m \le M$ colors $c_1,\ldots,c_m$, at most $\frac{\pi(\alpha)\pi(\beta)}{(\epsilon\Delta)^m}$ flow is routed into $\{\sigma \in \Phi_m : \sigma(i) = c_i, i \in [m]\}$. In other words, fix the colors $c_1,\ldots,c_m$. We want to bound the flow that passes through (into) the states of Φ_m, where vertices $1,\ldots,m$ are colored with $c_1,\ldots,c_m$ respectively. We do this by induction on m.

The base case: The total flow from α to β through Φ_i, for any i, is exactly $\pi(\alpha)\pi(\beta)$. For any $c_1 \in [k]$, at most $\frac{\pi(\alpha)\pi(\beta)}{\epsilon\Delta}$ flow is routed through $\{\sigma \in \Phi_1 : \sigma(1) = c_1\}$. This is because the last time vertex 1 changed color, at most $1/\epsilon\Delta$ of all the flow was routed to states where v's color is c_1.

The inductive step: From the inductive hypothesis, at most $\frac{\pi(\alpha)\pi(\beta)}{(\epsilon\Delta)^{m-1}}$ flow is routed through $\{\sigma \in \Phi_{m-1} : \sigma(i) = c_i, i \in [m-1]\}$. From the construction of the hybrid paths, for each of these states, at most $1/\epsilon\Delta$ of the flow entering it flows to a state where vertex m is colored c_m. ◀

Proof of Lemma 14

Proof. From the definition of the congestion on an edge (Equation (6)), we have

$$\begin{aligned}
\rho_f(t) &= \frac{1}{q(t)} \sum_{\alpha,\beta\in\Omega} \sum_{p:t\in p\in\mathcal{P}(\alpha,\beta)} f(p)|p| \\
&\le \frac{(\lambda+1)n\log n}{q(t)} \sum_{\alpha,\beta\in\Omega} \sum_{p:t\in p\in\mathcal{P}(\alpha,\beta)} f(p) && \text{(7a)} \\
&= 2|\Omega|k(\lambda+1)n^2\log n \sum_{\alpha,\beta\in\Omega} \sum_{p:t\in p\in\mathcal{P}(\alpha,\beta)} f(p) && \text{(7b)} \\
&= 2|\Omega|k(\lambda+1)n^2\log n \sum_{j=1}^{n} \sum_{\alpha,\beta\in\Omega} f_{j,t,\alpha,\beta} && \text{(7c)} \\
&= 2|\Omega|k(\lambda+1)n^2\log n \sum_{j=1}^{n} f_{j,t} \\
&\le 2|\Omega|k(\lambda+1)n^2\log n \sum_{j=1}^{n} \pi(\cdot)^2|\mathcal{C}_p| \cdot \left((1+\epsilon)\epsilon^{-1}\right)^{2|S_j|} && \text{(7d)} \\
&= \frac{2|\mathcal{C}_p|k(\lambda+1)n^2\log n}{|\Omega|} \sum_{j=1}^{n} \left((1+\epsilon)\epsilon^{-1}\right)^{2|S_j|} \\
&\le 2k(\lambda+1)n^3\log n \left((1+\epsilon)\epsilon^{-1}\right)^{2\,\mathrm{pw}(G)}.
\end{aligned}$$

Inequality (7a) is because the length of any hybrid path is at most $(\lambda+1)n\log n$; Equality (7b) is due to the definition of q: $q(\sigma,\sigma') = \pi(\sigma)P(\sigma,\sigma')$, where $\pi(\sigma) = |\Omega|^{-1}$ and $P(\sigma,\sigma') =$

$(2kn)^{-1}$; Equality (7c) is simply a rephrasing that holds because

$$\sum_{\alpha,\beta\in\Omega}\sum_{p:t\in p\in\mathcal{P}(\alpha,\beta)} f(p)$$

is the flow through t under f; Inequality (7d) is due to Lemma 13. The final inequality is due to Theorem 7, as the pathwidth of a graph equals its vertex separation number. ◀

Proof of Lemma 15

Proof. We use the function g' from singly-flawed to proper colorings described in Corollary 4 to define the flows that have a singly-flawed coloring as (at least) one of their endpoints. For every α, β such that $\alpha \in \mathcal{C}_{sf}$ and $\beta \in \mathcal{C}_p$, we route the entire flow on the transition $(\alpha, g'(\alpha))$ and then proceed using the canonical paths described above for routing the flow from $g'(\alpha)$ to β. If $\alpha \in \mathcal{C}_p$, and $\beta \in \mathcal{C}_{sf}$, we route the flow from α to $g'(\beta)$ using the canonical paths above and then on the edge $(g'(\beta), \beta)$. Finally, if $\alpha, \beta \in \mathcal{C}_{sf}$, we route the entire flow on $(\alpha, g'(\alpha))$, use the canonical paths above to route from $g'(\alpha)$ to $g'(\beta)$ and finally route the entire flow on $(g'(\beta), \beta)$. For every state $\sigma \in \mathcal{C}_p$, there are at most kn states $\sigma' \in \mathcal{C}_{sf} : g'(\sigma') = \sigma$. Therefore, we have multiplied the flow on every edge by at most

$$k^2n^2 + 2kn + 1 < 4k^2n^2, \tag{8}$$

where the first term is for pairs $\alpha, \beta \in \mathcal{C}_{sf}$, the third is for $\alpha, \beta \in \mathcal{C}_p$, and the second term on the left hand side is for mixed pairs. We added a further

$$kn\pi(\cdot)^2|\Omega| = \frac{2kn}{|\Omega|} < 1 \tag{9}$$

to each edge $(\sigma, g'(\sigma))$ and $(g'(\sigma), \sigma)$: there are at most $|\Omega|$ paths from a state $\sigma \in \mathcal{C}_p$ (to any other state), hence at most $kn|\Omega|$ paths from any $\sigma' \in \mathcal{C}_{sf}$. We absorb Inequality (9) and the fact that $\mathrm{pw}(G) + \log n = (\lambda + 1)\log n$ into Inequality (8). Multiplying the bound of Lemma 14 by $4k^2n^2$ gives the required bound. ◀

Proof of Theorem 16

Proof. We denote by $\hat{\pi}$ the distribution reached by $\mathcal{MC}$ after $\tau(\delta_1)$ steps. By definition, the total variation distance between and π and $\hat{\pi}$ is at most δ_1, hence for any $S \subset \Omega$, it holds that

$$|\pi(S) - \hat{\pi}(S)| \leq \delta_1. \tag{10}$$

Choosing $S = \mathcal{C}_p$ and applying Corollary 3 gives that the probability of the final state being a proper coloring is at least $\frac{1}{kn+1} - \delta_1$. Our choice of T is so that Hoeffding's bound guarantees that Algorithm 1 will output a proper coloring during the **for** loop (i.e., a final state of the Markov chain and not an arbitrary coloring), with probability at least $1 - \delta_H$, where $\delta_H \leq \frac{\delta}{3}$.

To show that Algorithm 1 is an almost uniform sampler for proper colorings, we need to show that the sampled coloring is drawn from a distribution that is close to uniform. In other words, if σ is the state output by Algorithm 1, then for any $S \subseteq \mathcal{C}_p$

$$\frac{\pi(S)}{\pi(\mathcal{C}_p)} - \delta \leq \Pr[\sigma \in S] \leq \frac{\pi(S)}{\pi(\mathcal{C}_p)} + \delta.$$

$$
\begin{aligned}
\Pr[\sigma \in S] &\geq \frac{\hat{\pi}(S)}{\hat{\pi}(\mathcal{C}_p)}(1-\delta_H) && \text{(11a)}\\
&\geq \frac{\hat{\pi}(S)}{\hat{\pi}(\mathcal{C}_p)} - \delta_H \\
&\geq \frac{\pi(S)-\delta_1}{\pi(\mathcal{C}_p)+\delta_1} - \delta_H && \text{(11b)}\\
&\geq \frac{\pi(S)-2\delta_1}{\pi(\mathcal{C}_p)} - \delta_H && \text{(11c)}\\
&= \frac{\pi(S)}{\pi(\mathcal{C}_p)} - \frac{2\delta_1}{\pi(\mathcal{C}_p)} - \delta_H \\
&\geq \frac{\pi(S)}{\pi(\mathcal{C}_p)} - \frac{2\delta}{3} - \frac{\delta}{3},
\end{aligned}
$$

where (11a) is the probability that the Algorithm outputs a proper coloring and it is in S; (11b) is due to Equation (10); (11c) is because $\frac{a-1}{b+1} \geq \frac{a+2}{b}$ for $b \geq a$.

The complementary $\Pr[\sigma \in S] \leq \frac{\pi(S)}{\pi(\mathcal{C}_p)} + \delta$ is immediate by considering the set $\mathcal{C}_p \setminus S$: if this were not the case then it would hold that $\Pr[\sigma \in S] + \Pr[\sigma \in \mathcal{C}_p \setminus S] > 1$. ◀

Explicit Strong LTCs with Inverse Poly-Log Rate and Constant Soundness

Michael Viderman
Yahoo Research, Haifa, Israel
viderman@oath.com

Abstract

An error-correcting code $C \subseteq \mathbb{F}^n$ is called (q, ϵ)-strong locally testable code (LTC) if there exists a tester that makes at most q queries to the input word. This tester accepts all codewords with probability 1 and rejects all non-codewords $x \notin C$ with probability at least $\epsilon \cdot \delta(x, C)$, where $\delta(x, C)$ denotes the relative Hamming distance between the word x and the code C. The parameter q is called the query complexity and the parameter ϵ is called soundness.

Goldreich and Sudan (J.ACM 2006) asked about the existence of strong LTCs with constant query complexity, constant relative distance, constant soundness and inverse polylogarithmic rate. They also asked about the explicit constructions of these codes.

Strong LTCs with the required range of parameters were obtained recently in the works of Viderman (CCC 2013, FOCS 2013) based on the papers of Meir (SICOMP 2009) and Dinur (J.ACM 2007). However, the construction of these codes was *probabilistic*.

In this work we show that codes presented in the works of Dinur (J.ACM 2007) and Ben-Sasson and Sudan (SICOMP 2005) provide the *explicit* construction of strong LTCs with the above range of parameters. Previously, such codes were proven to be weak LTCs. Using the results of Viderman (CCC 2013, FOCS 2013) we prove that such codes are, in fact, strong LTCs.

2012 ACM Subject Classification Theory of computation → Interactive proof systems

Keywords and phrases Error-Correcting Codes, Tensor Products, Locally Testable Codes

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.58

Related Version The full version of this paper appeared as [44].

Acknowledgements The author thanks Or Meir for raising the suggestion to obtain an explicit construction of strong LTCs by applying the arguments of [43, 42] to the codes of [11]. We would like to thank the anonymous reviewers for their valuable comments.

1 Introduction

Probabilistically Checkable Proof (PCP) systems [2, 3, 21] (a.k.a. Holographic Proofs [4]) are proof systems that allow efficient probabilistic verification of a claim by reading few symbols of the proof. The celebrated PCP theorem [2, 3] is one of the main breakthrough results in complexity theory. This theorem asserts that for every language in $\mathcal{NP}$ there exists a polynomial-time PCP verifier that queries the polynomial length proof in a constant number of locations. The verifier is guaranteed to always accept valid proofs of true statements, and to accept any claimed proof of false assertions with low probability. The theorem has found many applications in theoretical computer science, especially in establishing lower bounds for approximation algorithms [6, 5, 21, 29].

Informally, most of the PCP constructions were achieved using error-correcting codes, possessing nice properties. Let us first give some auxiliary definitions regarding error-correcting codes.

A code over a finite alphabet Σ is a subset $\mathcal{C} \subseteq \Sigma^n$. A linear code over a finite field $\mathbb{F}$ is a linear subspace $\mathcal{C} \subseteq \mathbb{F}^n$. In this case, n is the blocklength of the code $\mathcal{C}$, denoted by $\text{blocklength}(\mathcal{C})$. The dimension of a linear code $\mathcal{C}$, denoted by $\dim(\mathcal{C})$, is its dimension as a vector space and is equal to $\log_{|\mathbb{F}|} |\mathcal{C}|$. The dimension of a non-linear code $\mathcal{C}$ over the alphabet Σ is defined to be $\dim(\mathcal{C}) = \log_{|\Sigma|} |\mathcal{C}|$. The rate of a code $\mathcal{C}$, denoted by $\text{rate}(\mathcal{C})$, is defined to be $\frac{\dim(\mathcal{C})}{\text{blocklength}(\mathcal{C})} = \frac{\dim(\mathcal{C})}{n}$.

We define the distance between two words $x, y \in \mathbb{F}^n$ to be $\Delta(x, y) = |\{i \mid x_i \neq y_i\}|$ and the relative distance to be $\delta(x, y) = \frac{\Delta(x,y)}{n}$. The distance of $\mathcal{C}$ is defined by $\Delta(\mathcal{C}) = \min_{x \neq y \in \mathcal{C}} \Delta(x, y)$ and its relative distance is defined by $\delta(\mathcal{C}) = \frac{\Delta(\mathcal{C})}{n}$. The Hamming weight of a word $x \in \mathbb{F}^n$ is denoted by $|x|$ and defined to be $|x| = |\{i \in [n] \mid x_i \neq 0\}|$. We note that if $\mathcal{C}$ is linear then $\Delta(\mathcal{C}) = \min_{c \in \mathcal{C} \setminus \{0\}} \{|c|\}$. One is typically interested in codes whose distance is linear to the blocklength of $\mathcal{C}$, i.e., $\Omega(n)$.

For $x \in \mathbb{F}^n$ and $\mathcal{C} \subseteq \mathbb{F}^n$, let $\delta(x, \mathcal{C}) = \min_{y \in \mathcal{C}} \{\delta(x, y)\}$ denote the relative distance of x from the code $\mathcal{C}$. If $\delta(x, \mathcal{C}) \geq \rho$, we say that x is ρ-far from $\mathcal{C}$ and otherwise x is ρ-close to $\mathcal{C}$.

Given a field $\mathbb{F}$, we say that there exists an *explicit* construction of a family of linear codes $\{C_n \subseteq \mathbb{F}^n\}_{n \in \mathbb{N}^+}$ if there exists an algorithm that on infinitely many inputs $n \in \mathbb{N}^+$ outputs the generating matrix of C_n.

1.1 Locally Testable Codes

Most of the PCP constructions (e.g., [7, 11, 18, 27]) are tightly related to a special kind of error-correcting codes possessing some testability properties. These codes are called *locally testable*.

In other words, locally testable codes (LTCs) are error correcting codes that have a tester, which is a randomized algorithm with oracle access to the received word x. The tester reads a sublinear amount of information from x and based on this "local view" decides if $x \in C$ or not. It should accept codewords with probability one, and reject words that are far (in Hamming distance) from the code with noticeable probability. Such codes are of interest in computer science due to their numerous connections to probabilistically checkable proofs (PCPs) and property testing (see the surveys [39, 24] for more information). LTCs were implicit already in [4] (cf. [24, Sec. 2.4]) and they were explicitly studied by Goldreich and Sudan [27] (see also [22, 38]).

By now several different constructions of LTCs are known including codes based on low-degree polynomials over finite fields and affine-invariant codes [1, 2, 16, 9, 8, 15, 28, 31, 33, 30, 37], constructions based on PCPs of proximity/assignment testers [7, 19, 18][1], sparse random linear codes [14, 32, 35] and tensor products of codes [10, 12, 13, 20, 36, 41, 34].

Basically, there are two kinds of LTCs: weak and strong. A code $\mathcal{C}$ is said to be (q, ϵ, ρ)-weak LTC if there exists a randomized algorithm T, called tester, that makes at most q queries to the input word w. If $w \in \mathcal{C}$ then T accepts w with probability 1, but if w is ρ-far from C the tester T rejects w with probability at least ϵ. Let us notice that the tester is not required to reject when $0 < \delta(w, \mathcal{C}) < \rho$. This is the reason why such codes are called *weak* LTCs.

In contrast to weak LTCs, the testers for strong LTCs are required to reject all non-codewords with corresponding probability. More formally, a code $\mathcal{C}$ is called (q, ϵ)-strong

[1] As was pointed out in [27], not all PCP constructions are known to yield LTCs, but some of them (e.g., PCPs of proximity/assignment testers) can be adapted to yield LTCs.

LTC if there exists a tester T that makes at most q queries to the input word w. If $w \in \mathcal{C}$ then T accepts w with probability 1, but if $w \notin \mathcal{C}$ then T rejects w with probability at least $\epsilon \cdot \delta(w, \mathcal{C})$. The parameter q is called the query complexity and the parameter ϵ is called soundness.

Informally, we say that a code $\mathcal{C}$ is a weak LTC if it has a linear distance and there exist constants $q, \epsilon > 0$ and $\rho \leq \delta(\mathcal{C})/3$ such that C is a (q, ϵ, ρ)-weak LTC. [2] Similarly, we say that a code $\mathcal{C}$ is a strong LTC if it has a linear distance and there exist constants $q, \epsilon > 0$ such that $\mathcal{C}$ is a (q, ϵ)-strong LTC.

LTCs were explicitly studied in the work of Goldreich and Sudan [27], who presented probabilistic construction of strong LTCs. These LTCs achieve constant query complexity, constant soundness and rate $\frac{1}{\exp(\tilde{O}(\sqrt{\log n}))}$, where n denotes the blocklength.

Later, other constructions of LTCs [11, 18, 36] succeeded to obtain the rate $\frac{1}{\text{polylog}(n)}$ together with constant query complexity and soundness, however these codes were weak LTCs. It can be verified that every strong LTC is also a weak LTC, but some weak LTCs are not strong LTCs [43]. So, strong LTCs are strictly stronger objects than weak LTCs. As was pointed out by Goldreich [23], strong LTCs correspond to proximity oblivious testers [26] whereas weak LTCs are even weaker than ordinary testers, i.e., the testers for weak LTCs are supposed to work only for a fixed value of the proximity parameter. In the journal version of [27], the authors pointed out that all known LTCs that achieve inverse polylogarithmic rate are weak LTCs, and asked about the existence of strong LTCs with polylogarithmic rate and, in particular, about the *explicit* construction of such codes [27, Section 6].

The previous papers of the author [43, 42] showed a *probabilistic* construction of binary linear 3-query strong LTCs with inverse polylogarithmic rate, constant soundness and constant relative distance. In this paper (Section 1.4), we show the explicit construction of linear strong LTCs with constant query complexity, constant soundness, polylogarithmic rate and constant relative distance over a fixed field, therefore resolving a question raised by Goldreich and Sudan [27].[3] We would like to stress that the codes we refer to were, in fact, only a special case of PCPs and PCPs of proximity constructed in [11, 18]. Therefore, this work discovers strong local testability properties in the objects tightly related to the short PCPs and so, might be useful for the future PCPs related applications.

As was mentioned previously, we prove that the codes of [11, 18] yield explicit strong LTCs. To do that we want to reuse the arguments of [43, 42] to prove that the codes of [11, 18] are explicit strong LTCs. These codes (as well as codes of [36, 43, 42]) involve two kind of symbols: code symbols and proof symbols (called also core symbols and non-core symbols, respectively, in [43]). [4] However, the codes of [11] have good distance only on the code symbols with no guarantee on the proof symbols, while the arguments used in [43] require both good distance on the code coordinates and on the proof coordinates. Therefore, our main technical ingredient in this work is observing that the results of [43] can be reproved with only requirement of good distance on the code coordinates.

---

[2] The parameter ρ is required to be less than $\delta(\mathcal{C})/2$ to avoid trivial solutions like claiming that every perfect code $\mathcal{C}$ is a $(0, 1, \delta(\mathcal{C})/2)$-weak LTC. Recall that a code $\mathcal{C} \subseteq \mathbb{F}^n$ is called perfect if there are no words in $\mathbb{F}^n$ that are $(\delta(\mathcal{C})/2)$-far from $\mathcal{C}$. So, in this case one could say that no queries are needed and all $(\delta(\mathcal{C})/2)$-far words are rejected with probability 1 vacuously.

[3] A suggestion to show such explicit construction was raised in personal discussion with Or Meir, and later was asked in [42].

[4] The concepts like 'code' and 'proof' coordinates appear in PCP related literature, but might appear under different names (statement and proof) and with slightly different meaning as well.

The rest of the paper is organized as follows. We provide the necessary definitions in Section 1.2 and state our main result (Theorem 4) in Section 1.4. The main ideas and the overview of the proof of Theorem 4 are given in Section 2. The proof of Theorem 4 is postponed to Section A.

1.2 Preliminaries

Let $[n]$ be the set $\{1,\ldots,n\}$. For $w \in \mathbb{F}^n$, let $\mathrm{supp}(w) = \{i \in [n] \mid w_i \neq 0\}$ and $|w| = |\mathrm{supp}(w)|$. For $u = (u_1, u_2, \ldots, u_n)$, $v = (v_1, v_2, \ldots, v_n) \in \mathbb{F}^n$ let $\langle u, v\rangle$ denote the bilinear function from $\mathbb{F}^n \times \mathbb{F}^n$ to $\mathbb{F}$ defined by $\langle u, v\rangle = \sum_{i=1}^{n} u_i v_i$. The dual code is defined by $\mathcal{C}^\perp = \{u \in \mathbb{F}^n \mid \forall c \in \mathcal{C} : \langle u, c\rangle = 0\}$. Similarly, we define $\mathcal{C}^\perp_{\leq t} = \left\{u \in \mathcal{C}^\perp \mid |u| \leq t\right\}$. For $w \in \mathbb{F}^n$ and $S = \{j_1, j_2, \ldots, j_m\} \subseteq [n]$ we let $w|_S = (w_{j_1}, w_{j_2}, \ldots, w_{j_m})$, where $j_1 < j_2 < \ldots < j_m$, be the restriction of w to the subset S. Similarly, we let $\mathcal{C}|_S = \{c|_S \mid c \in \mathcal{C}\}$ denote the projection of the code $\mathcal{C}$ onto S. We define $\mathcal{C}|_{-S} = \mathcal{C}|_{[n]\setminus S}$, i.e., projection of the code $\mathcal{C}$ to all coordinates besides S. For $A \subseteq \mathbb{N}$ and $b \in \mathbb{N}$ we let $A + b = b + A = \{a + b \mid a \in A\}$. For a code $\mathcal{C}$ we let $\mathrm{coord}(\mathcal{C})$ to be a coordinate set of the code, e.g., if $\mathcal{C} \subseteq \mathbb{F}^n$ then $\mathrm{coord}(\mathcal{C}) = [n]$.

For the distribution $\mathcal{D}$ over the subsets of $[n]$ we let $\mathcal{D}(I)$ to denote the probability that a subset $I \subseteq [n]$ is selected by $\mathcal{D}$ and $\mathrm{supp}(\mathcal{D}) = \{I \subseteq [n] \mid \mathcal{D}(I) > 0\}$. For $i \in [n]$ we let $N_{\mathcal{D}}(i) = \{I \in \mathrm{supp}(\mathcal{D}) \mid i \in I\}$.

Now we define testers and LTCs (see [27, 43] for the justification of this definition).

▶ **Definition 1** (LTCs and Testers)**.** A q-query tester for a code $\mathcal{C} \subseteq \mathbb{F}^n$ is a distribution $\mathcal{D}$ over subsets $I \subseteq [n]$ such that $|I| \leq q$. A q-query tester $\mathcal{D}$ is a (q, ϵ, ρ)-weak tester if for all $w \in \mathbb{F}^n$, $\delta(w, \mathcal{C}) \geq \rho$ we have $\mathop{\mathbf{Pr}}_{I\sim\mathcal{D}}[w|_I \notin \mathcal{C}|_I] \geq \epsilon$. A q-query tester $\mathcal{D}$ is a (q, ϵ)-strong tester if for all $w \in \mathbb{F}^n$ we have $\mathop{\mathbf{Pr}}_{I\sim\mathcal{D}}[w|_I \notin \mathcal{C}|_I] \geq \epsilon \cdot \delta(w, \mathcal{C})$.

A code $\mathcal{C} \subseteq \mathbb{F}^n$ is a (q, ϵ, ρ)-*weak LTC* if it has a (q, ϵ, ρ)-weak tester. A code $\mathcal{C} \subseteq \mathbb{F}^n$ is a (q, ϵ)-*strong LTC* if it has a (q, ϵ)-strong tester.

▶ Remark. Although the tester in Definition 1 does not output accept or reject, the way a standard tester does, it can be converted to output accept, reject as follows. Whenever the task is to test whether $w \in \mathcal{C}$ and a subset $I \subseteq [n]$ is selected by the tester, the tester can output accept if $w|_I \in \mathcal{C}|_I$ and otherwise output reject. In this manner, the tester always accepts the codewords of $\mathcal{C}$.

It is not hard to see that every strong LTC is a weak LTC, but not vice versa [43, Proposition B.1].

The support of the tester $\mathcal{D}_C$ for the code $\mathcal{C} \subseteq \mathbb{F}^n$ is denoted by $\mathrm{supp}(\mathcal{D}_C)$ and defined to be $\mathrm{supp}(\mathcal{D}_C) = \{I \subseteq [n] \mid \mathcal{D}_C(I) > 0\}$. We say that $\mathcal{D}_C$ is uniform over its support if for each $I_1, I_2 \in \mathrm{supp}(\mathcal{D}_C)$ we have $\mathcal{D}_C(I_1) = \mathcal{D}_C(I_2)$. The test neighbors of the coordinate $i \in [n]$ are defined by $N_{\mathcal{D}_C}(i) = \{I \subseteq [n] \mid \mathcal{D}_C(I) > 0, i \in I\}$.

1.2.1 Tensor Product of Codes

The definitions appearing here are standard in the literature on tensor-based LTCs (e.g., [10, 36, 43, 20, 41, 12]). For $x \in \mathbb{F}^{n_1}$ and $y \in \mathbb{F}^{n_2}$ we let $x \otimes y$ denote the tensor product of x and y (i.e., the matrix M with entries $M(i, j) = x_j \cdot y_i$ where $(i, j) \in [n_2] \times [n_1]$). Let $\mathcal{R} \subseteq \mathbb{F}^{n_1}$ and $\mathcal{C} \subseteq \mathbb{F}^{n^2}$ be linear codes. We define the tensor product code $\mathcal{R} \otimes \mathcal{C}$ to be the linear space spanned by words $r \otimes c \in \mathbb{F}^{n_2 \times n_1}$ for $r \in \mathcal{R}$ and $c \in \mathcal{C}$. Some known facts regarding the tensor products (see e.g., [20]):

- The code $\mathcal{R} \otimes \mathcal{C}$ consists of all $n_2 \times n_1$ matrices over F whose rows belong to $\mathcal{R}$ and columns belong to $\mathcal{C}$,
- $\dim(\mathcal{R} \otimes \mathcal{C}) = \dim(\mathcal{R}) \cdot \dim(\mathcal{C})$,
- $\text{rate}(\mathcal{R} \otimes \mathcal{C}) = \text{rate}(\mathcal{R}) \cdot \text{rate}(\mathcal{C})$ and
- $\delta(\mathcal{R} \otimes \mathcal{C}) = \delta(\mathcal{R}) \cdot \delta(\mathcal{C})$.

We let $\mathcal{C}^{\otimes 1} = \mathcal{C}$ and $\mathcal{C}^{\otimes m} = \mathcal{C}^{\otimes(m-1)} \otimes \mathcal{C}$ for $m > 1$. Note by this definition, $\mathcal{C}^{\otimes 2^0} = \mathcal{C}$ and $\mathcal{C}^{2^m} = \mathcal{C}^{\otimes 2^{m-1}} \otimes \mathcal{C}^{\otimes 2^{m-1}}$ for $t > 0$. We also notice that for a code $\mathcal{C} \subseteq \mathbb{F}^n$ and $m \geq 1$ it holds that $\text{rate}(\mathcal{C}^{\otimes m}) = (\text{rate}(\mathcal{C}))^m$, $\delta(\mathcal{C}^{\otimes m}) = (\delta(\mathcal{C}))^m$ and the blocklength of $C^{\otimes m}$ is n^m. We notice that if $\text{coord}(\mathcal{C}) = [n]$ then the coordinate set of $\mathcal{C} \otimes \mathcal{C}$ is $\text{coord}(\mathcal{C} \otimes \mathcal{C}) = [n] \times [n]$.

1.3 Robust Testing

In this section we define some properties of codes that are sufficient for robust testing. We start this section by defining the notion of robustness (Definition 3) as was introduced in [10] following [7]. To do that we provide the definition of local distance (Definition 2), which will be used in Definition 3 and later in our proofs. In this section we use n to denote the blocklength of the code C, i.e., $n = |\text{coord}(C)|$. Without loss of generality we assume that $\text{coord}(C) = [n]$.

▶ **Definition 2** (Local distance). Let $C \subseteq \mathbb{F}^n$ be a code and $w|_I$ be the view on the coordinate set $I \subseteq [n]$ obtained from the word $w \in \mathbb{F}^n$. The local distance of w from C with respect to I is $\Delta(w|_I, C|_I) = \min c \in C\{\Delta(w|_I, c|_I)\}$ and similarly the relative local distance of w from C with respect to I is $\delta(w|_I, C|_I) = \min_{c \in C} \{\delta(w|_I, c|_I)\}$.

Informally, we say that a tester is robust if for every word that is far from the code, the tester view is far on average from any consistent view. This notion was defined for LTCs following an analogous definition for PCPs [7, 18]. We are ready to provide a general definition of robustness.

▶ **Definition 3** (Robustness). Given a tester (i.e., a distribution) $\mathcal{D}$ for the code $C \subseteq \mathbb{F}^n$, we let

$$\rho^{\mathcal{D}}(w) = \mathop{\mathbf{E}}_{I \sim \mathcal{D}}[\delta(w|_I, C|_I)] \text{ be the expected relative local distance of input } w.$$

We say that the tester D has robustness $\rho^{\mathcal{D}}(C)$ on the code C if for every $w \in \mathbb{F}^n$ it holds that $\rho^{\mathcal{D}}(w) \geq \rho^{\mathcal{D}}(C) \cdot \delta(w, C)$. Let $\{C_n\}_n$ be a family of codes where C_n is of blocklength n and D_n is a tester for C_n. A family of codes $\{C_n\}_n$ is robustly testable with respect to testers $\{D_n\}_n$ if there exists a constant $\alpha > 0$ such that for all n we have $\rho^{\mathcal{D}_n}(C_n) \geq \alpha$.

1.4 Main Result

In this paper we show the *explicit* construction of strong LTCs over a fixed field with a range of parameters asked by Goldreich and Sudan [27]. Although the requested range of parameters was achieved for the *probabilistic* construction of strong LTCs [43, 42], explicit strong LTCs with this range of parameters was not obtained.

▶ **Theorem 4** (Main Theorem). *There exist constants $q, d, \epsilon, \gamma > 0$ and a constant size field $\mathbb{F}$ such that for infinitely many $n \in \mathbb{N}^+$ we have an* explicit *construction of a linear code $C \subseteq \mathbb{F}^n$, where*

- *C is a (q, ϵ)-strong LTC,*
- *$\delta(C) \geq \gamma$ and $\text{rate}(C) \geq \frac{1}{\log^d n}$.*

The proof of Theorem 4 is given in Section A. Before we are going over the proof let us present the overview and the main ideas of this proof in Section 2.

2 Overview of the Proof

Roughly speaking, to prove Theorem 4 we apply the arguments of [43, 42] to the construction of [18] (which was obtained by applying the gap amplification procedure on the codes of [11]). However, it is impossible to use the observations of [43, 42] as is since there are considerable differences between the construction of [11] and the construction of [43, 42] (based on [36]).

The first obstacle is that the codes of [11] were obtained from iterative polynomial constructions, while the codes appeared in [43, 42] (based on [36]) are tensor products of general error correcting codes. This issue is resolved since the construction of [11] can be viewed as a kind of tensoring over a large field (see [36, Section 7.2]). It is worth to mention that while [43, 36] used iterative 3-wise tensor products for the code construction, [11] can be viewed as iterative 2-wise tensor products (see Section A.2). In general, codes composed as 2-wise tensor products might be not testable [17, 40, 25], however, there are still ways to use such kind of code products to obtain locally testable codes, e.g., [20, 12, 13, 36].

The second difficulty is that both kind of constructions have the 'code' coordinates and 'proof' coordinates. While there is no distance guarantee on the proof coordinates in [11], the constant relative distance on these coordinates is required in the works of [43, 42]. More formally, consider a code $C \subseteq \mathbb{F}^n$ from [11] and assume that $[n]$ is partitioned to code coordinates set s_1 and proof coordinates set s_2, i.e., $s_1 \cup s_2 = [n]$, $s_1 \cap s_2 = \emptyset$. So, while nothing could be guaranteed regarding $\delta(C|_{s_2})$, to use the arguments of [43, 42] we need to know that $\delta(C|_{s_2})$ is constant, or in words, $C|_{s_2}$ has constant relative distance.

In [43] there was suggested a way to slightly modify the construction of [36] to ensure the good distance on the proof coordinates. However, it would be much more problematic to modify a part of coordinates in the construction of [11] since it has very concrete and non-flexible polynomial-based structure. Therefore, the main technical ingredient in our work is the modification of the arguments of [43, 42] to avoid 'good distance' requirement from the proof coordinates. We succeed to prove that it was redundant requirement and only a good distance on the code coordinates is sufficient.

So far, the proof (presented in Section A) is starting from presenting central concepts which play crucial role in the proof. After these concepts are presented, the rest of the proof is done in 3 stages. The first stage (Section A.2) argues that a construction of [11] can be viewed as iterative tensoring with some "smart" projection procedure, where in every iteration a tensoring is done only over the code coordinates and a part of the symbols are moved from the code coordinates part to the proof coordinates part.

The second stage (Section A.3) is to recall the arguments of [43] to argue that codes of [11] are COLTCs (Definition 7). This stage reproves one of the technical ingredients of [43] without "good" distance requirement from the proof coordinates, and in particular, Corollary 16 shows that the codes of [11] are COLTCs (and hence strong LTCs) even though no distance is guaranteed on the proof coordinates.

Finally, in Section A.4 we recall the arguments of [42] to argue that when the gap amplification procedure of [18] is applied to the codes of [11] it yields strong LTCs. In words, this section recalls a relaxed LTC (rLTC) concept (Definition 17) from [42], and the observation that strong LTCs are also relaxed LTCs with the corresponding range parameters. Theorem 19 and its Corollary 20 claim that the gap amplification of Dinur [18] can be applied to the codes of [11], proven to be COLTCs in Corollary 16 (and thus are strong LTCs and rLTCs), and yields rLTCs with the constant first soundness parameter and inverse poly-log second soundness parameter. Corollary 21 implies that such rLTCs can be explicitly converted to the required strong LTCs with constant soundness.

The fact that the codes construction of [11] and the gap amplification procedure are deterministic, yields *explicit* strong LTCs.

Theorem 4 will follow from Corollary 20 and Corollary 21.

References

1. Noga Alon, Tali Kaufman, Michael Krivelevich, Simon Litsyn, and Dana Ron. Testing Reed-Muller codes. *IEEE Transactions on Information Theory*, 51(11):4032–4039, 2005. doi:10.1109/TIT.2005.856958.
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, 1998.
3. Sanjeev Arora and Shmuel Safra. Probabilistic Checking of Proofs: A New Characterization of NP. *Journal of the ACM*, 45(1):70–122, 1998.
4. László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking Computations in Polylogarithmic Time. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing (STOC), May 5-8, 1991, New Orleans, Louisiana, USA*, pages 21–31. ACM, 1991. doi:10.1145/103418.103428.
5. M. Bellare, S. Goldwasser, C. Lund, and A. Russell. Efficient probabilistically checkable proofs and applications to approximation. In *Proceedings of the 25th Annual ACM Symposium on Theory of Computing (STOC)*, pages 294–304, New York, 1993. ACM SIGACT, ACM Press.
6. Mihir Bellare, Oded Goldreich, and Madhu Sudan. Free Bits, PCPs, and Nonapproximability—Towards Tight Results. *SIAM Journal on Computing*, 27(3):804–915, 1998.
7. 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.
8. Eli Ben-Sasson, Elena Grigorescu, Ghid Maatouk, Amir Shpilka, and Madhu Sudan. On sums of locally testable affine invariant properties. In *Proceedings of Approximation, Randomization, and Combinatorial Optimization (APPROX-RANDOM)*, volume 6845 of *Lecture Notes in Computer Science*, pages 400–411. Springer, 2011. doi:10.1007/978-3-642-22935-0.
9. Eli Ben-Sasson, Noga Ron-Zewi, and Madhu Sudan. Sparse affine-invariant linear codes are locally testable. In *FOCS*, pages 561–570. IEEE Computer Society, 2012. URL: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6374356.
10. 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.
11. Eli Ben-Sasson and Madhu Sudan. Short PCPs with polylog query complexity. *SIAM J. Comput*, 38(2):551–607, 2008. doi:10.1137/050646445.
12. Eli Ben-Sasson and Michael Viderman. Composition of Semi-LTCs by Two-Wise Tensor Products. In *Proceedings of Approximation, Randomization, and Combinatorial Optimization (APPROX-RANDOM)*, volume 5687 of *Lecture Notes in Computer Science*, pages 378–391. Springer, 2009. doi:10.1007/978-3-642-03685-9.
13. Eli Ben-Sasson and Michael Viderman. Tensor Products of Weakly Smooth Codes are Robust. *Theory of Computing*, 5(1):239–255, 2009. doi:10.4086/toc.2009.v005a012.
14. Eli Ben-Sasson and Michael Viderman. Low rate is insufficient for local testability. In *Proceedings of Approximation, Randomization, and Combinatorial Optimization (APPROX-RANDOM)*, volume 6302 of *Lecture Notes in Computer Science*, pages 420–433. Springer, 2010. doi:10.1007/978-3-642-15369-3.

15 Arnab Bhattacharyya, Swastik Kopparty, Grant Schoenebeck, Madhu Sudan, and David Zuckerman. Optimal testing of reed-muller codes. In *FOCS*, pages 488–497. IEEE Computer Society, 2010. doi:10.1109/FOCS.2010.54.

16 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.

17 Don Coppersmith and Atri Rudra. On the Robust Testability of Product of Codes. *Electronic Colloquium on Computational Complexity (ECCC)*, TR05-104, 2005. URL: http://eccc.hpi-web.de/eccc-reports/2005/TR05-104/index.html.

18 Irit Dinur. The PCP theorem by gap amplification. *Journal of the ACM*, 54(3):12:1–12:44, jun 2007.

19 Irit Dinur and Omer Reingold. Assignment Testers: Towards a Combinatorial Proof of the PCP Theorem. *SIAM Journal on Computing*, 36(4):975–1024, 2006. doi:10.1137/S0097539705446962.

20 Irit Dinur, Madhu Sudan, and Avi Wigderson. Robust Local Testability of Tensor Products of LDPC Codes. In *Proceedings of Approximation, Randomization, and Combinatorial Optimization (APPROX-RANDOM)*, volume 4110 of *Lecture Notes in Computer Science*, pages 304–315. Springer, 2006. doi:10.1007/11830924_29.

21 Uriel Feige, Shafi Goldwasser, László Lovász, Shmuel Safra, and Mario Szegedy. Interactive proofs and the hardness of approximating cliques. *Journal of the ACM*, 43(2):268–292, 1996. doi:10.1145/226643.226652.

22 Katalin Friedl and Madhu Sudan. Some improvements to total degree tests. In *Third Israel Symposium on Theory of Computing and Systems, ISTCS 1995, Tel Aviv, Israel, January 4-6, 1995, Proceedings*, pages 190–198, 1995. doi:10.1109/ISTCS.1995.377032.

23 Oded Goldreich. Home page. http://www.wisdom.weizmann.ac.il/~oded/. URL: http://dl.acm.org/author_page.cfm?id=81336489395.

24 Oded Goldreich. Short Locally Testable Codes and Proofs (Survey). *Electronic Colloquium on Computational Complexity (ECCC)*, 2005. URL: http://eccc.hpi-web.de/eccc-reports/2005/TR05-014/index.html.

25 Oded Goldreich and Or Meir. The tensor product of two good codes is not necessarily robustly testable. *Inf. Process. Lett*, 112(8-9):351–355, 2012. doi:10.1016/j.ipl.2012.01.007.

26 Oded Goldreich and Dana Ron. On proximity-oblivious testing. *SIAM J. Comput*, 40(2):534–566, 2011. doi:10.1137/100789646.

27 Oded Goldreich and Madhu Sudan. Locally testable codes and PCPs of almost-linear length. *Journal of the ACM*, 53(4):558–655, 2006. doi:10.1145/1162349.1162351.

28 Elena Grigorescu, Tali Kaufman, and Madhu Sudan. Succinct representation of codes with applications to testing. *SIAM J. Discrete Math*, 26(4):1618–1634, 2012. doi:10.1137/100818364.

29 Johan Håstad. Some optimal inapproximability results. *Journal of the ACM*, 48(4):798–859, 2001. doi:10.1145/502090.502098.

30 Tali Kaufman and Shachar Lovett. New Extension of the Weil Bound for Character Sums with Applications to Coding. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, (FOCS)*, pages 788–796. IEEE, 2011. URL: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6108120.

31 Tali Kaufman and Dana Ron. Testing polynomials over general fields. *SIAM J. Comput*, 36(3):779–802, 2006. doi:10.1137/S0097539704445615.

32 Tali Kaufman and Madhu Sudan. Sparse Random Linear Codes are Locally Decodable and Testable. In *FOCS*, pages 590–600. IEEE Computer Society, 2007. doi:10.1109/FOCS.2007.65.

33 Tali Kaufman and Madhu Sudan. Algebraic property testing: the role of invariance. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC), Victoria, British Columbia, Canada, May 17-20, 2008*, pages 403–412. ACM, 2008. doi:10.1145/1374376.1374434.

34 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. doi:10.1145/3051093.

35 Swastik Kopparty and Shubhangi Saraf. Local list-decoding and testing of random linear codes from high error. In *Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC)*, pages 417–426. ACM, 2010. doi:10.1145/1806689.1806748.

36 Or Meir. Combinatorial Construction of Locally Testable Codes. *SIAM J. Comput*, 39(2):491–544, 2009. doi:10.1137/080729967.

37 Noga Ron-Zewi and Madhu Sudan. A new upper bound on the query complexity for testing generalized reed-muller codes. In *Proceedings of Approximation, Randomization, and Combinatorial Optimization (APPROX-RANDOM)*, volume 7408 of *Lecture Notes in Computer Science*, pages 639–650. Springer, 2012. doi:10.1007/978-3-642-32512-0.

38 Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM J. Comput.*, 25(2):252–271, 1996. doi:10.1137/S0097539793255151.

39 Luca Trevisan. Some Applications of Coding Theory in Computational Complexity, 2004. URL: http://arxiv.org/abs/cs/0409044.

40 Paul Valiant. The Tensor Product of Two Codes Is Not Necessarily Robustly Testable. In *Proceedings of Approximation, Randomization, and Combinatorial Optimization (APPROX-RANDOM)*, volume 3624 of *Lecture Notes in Computer Science*, pages 472–481. Springer, 2005. doi:10.1007/11538462_40.

41 Michael Viderman. A combination of testability and decodability by tensor products. In *Proceedings of Approximation, Randomization, and Combinatorial Optimization (APPROX-RANDOM)*, volume 7408 of *Lecture Notes in Computer Science*, pages 651–662. Springer, 2012. doi:10.1007/978-3-642-32512-0.

42 Michael Viderman. Strong LTCs with inverse poly-log rate and constant soundness. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS, Berkeley, CA, USA, 26-29 October, 2013*, pages 330–339, 2013.

43 Michael Viderman. Strong LTCs with inverse polylogarithmic rate and soundness. In *Proceedings of the 28th Conference on Computational Complexity, CCC, Palo Alto, California, USA, 5-7 June, 2013*, pages 255–265, 2013.

44 Michael Viderman. Explicit strong ltcs with inverse poly-log rate and constant soundness. *Electronic Colloquium on Computational Complexity (ECCC)*, 22:20, 2015. URL: http://eccc.hpi-web.de/report/2015/020.

A Proof of Theorem 4

In this section we present the proof of Theorem 4 which contains three parts. Let us first present in Section A.1 the central concepts from [43] which play the important part in this work as well.

A.1 Preliminary notations: Core Oriented LTCs and Core Oriented Robustness

▶ **Definition 5** (A core of the code). Let $C \subseteq \mathbb{F}^n$ be a linear code. A core of the code C, denoted by $A(C)$, is a nonempty subset of $[n]$ such that $\dim(C) = \dim(C|_{A(C)})$, i.e., any assignment to the entries of $A(C)$ uniquely determines the entries of $[n] \setminus A(C)$. In particular, for any $c \in C$ there is no $c' \in C$ such that $c|_{A(C)} = c'|_{A(C)}$ and $c|_{[n]\setminus A(C)} \neq c'|_{[n]\setminus A(C)}$.

We say that $A(C)$ is a γ-core of the code C if $A(C)$ is a core of C, $\delta(C|_{A(C)}) = \frac{\Delta(C|_{A(C)})}{|A(C)|} \geq \gamma$.

Clearly, there might be many options for $A(C)$, and in this case we fix only one such option. If $A(C) = [n]$ then for any $w, w' \in \mathbb{F}^n$ we let $\delta(w|_{[n]\setminus A(C)}, w'|_{[n]\setminus A(C)}) = \delta(w|_{[n]\setminus A(C)}, C|_{[n]\setminus A(C)}) = 0$.

We let residue(C) = [n] \ A(C) to be the non-core coordinates of C.

Usually, in the locally testable codes the distance is measured exactly as in the general error-correcting codes, i.e., with respect to the entire blocklength. However, when we consider a specific subset of coordinates, called the core of the code, we need to define a new concept of distance (used in [43]).

▶ **Definition 6** (Core oriented distance)**.** Assume $C \subseteq \mathbb{F}^n$ is a linear code and $A(C)$ is its core. We define a core oriented distance between two words $w, w' \in F^n$ to be

$$\delta_{A(C)}(w, w') = \max\left\{\delta(w, w'), \delta(w|_{A(C)}, w'|_{A(C)})\right\},$$

and a core oriented distance between the word $w \in F^n$ and the code C to be

$$\delta_{A(C)}(w, C) = \min_{c \in C}(\delta_{A(C)}(w, c)).$$

We note that for every code $C \subseteq \mathbb{F}^n$ with a core $A(C)$ and $w \in F^n$ it holds that $\delta_{A(C)}(w, C) \geq \delta(w, C)$. In particular, if w is δ-close to C with respect to the core oriented distance then w is δ-close to C with respect to the "standard" distance $\delta(w, C)$.

An important building block in the proof is a new kind of local testable code defined in [43].

▶ **Definition 7** (Core Oriented LTC (COLTC))**.** Let $C \subseteq \mathbb{F}^n$ be a linear code and let $\mathcal{D}$ be a distribution over subsets $I \subseteq [n]$ such that $|I| \leq q$. A $\mathcal{D}$ is a (q, ϵ)-COLTC tester if, given that $A(C)$ is a core of C, for all $w \in \mathbb{F}^n$ we have

$$\mathop{\mathbf{Pr}}_{I \sim D}[w|_I \notin C|I] \geq \epsilon \cdot \delta_{A(C)}(w, C).$$

A code $C \subseteq \mathbb{F}^n$ is called a (q, ϵ)-COLTC if it has a (q, ϵ)-COLTC tester.

Let C be a linear code and $A(C)$ be its core. If C is a (q, ϵ)-COLTC (with respect to the tester D_C) then C is a (q, ϵ)-strong LTC. To see this let $w \in \mathbb{F}^n$ and note that

$$\mathop{\mathbf{Pr}}_{I \sim D_C}[w|_I \in C|_I] \geq \epsilon \cdot \delta_{A(C)}(w, c) \geq \epsilon \cdot \delta(w, C).$$

A.1.1 Core Oriented Robust Testing of COLTCs

Now we present one of the central concepts in [43] called "core robustness". Before that recall the more standard notion of robust testing (Definition 3) used e.g., in [10, 20, 12, 36].

In contrast to robust testing, in a core oriented robust testing (Definition 8) we pay a special attention on the core of the code and consider a core oriented distance rather than a standard distance as in Definition 3.

▶ **Definition 8** (Core robustness)**.** Assume that $\mathcal{D}$ is a tester (i.e., a distribution) for the linear code $C \subseteq \mathbb{F}^n$ with a core $A(C)$. Assume that for every subset $I \subseteq [n]$ such that $\mathcal{D}(I) > 0$ it holds that $C|_I$ is a linear code with a core $A(C|_I)$. We let

$$\rho^{\mathcal{D}}_{A(C)}(w) = \mathop{\mathbf{E}}_{I \sim \mathcal{D}}\left[\delta_{A(C|_I)}(w|_I, C|_I)\right]$$

be the expected core oriented relative local distance of input w. We say that the tester $\mathcal{D}$ for the code C has core robustness $\rho^{\mathcal{D}}_{A(C)}(C)$ if for every $w \in \mathbb{F}^n$ it holds that $\rho^{\mathcal{D}}_{A(C)}(w) \geq \rho^{D}_{A(C)}(C) \cdot \delta_{A(C)}(w, C)$.

It turns out that a combination of "core robustness" with COLTCs is highly useful.

▶ **Claim 9** (Claim 7.2 in [43]). *Let C be a (q, ϵ)-COLTC and let D_C be its tester. Let $\hat{C} \subseteq \mathbb{F}^{\mathrm{coord}(\hat{C})}$ be a linear code with a core $A(\hat{C})$ and let $\mathcal{D}_{\hat{C}}$ be its tester. Assume that $\rho^{D_{\hat{C}}}_{A(\hat{C})}(\hat{C}) \geq \alpha$ and for every local view $I \subseteq \mathrm{coord}(\hat{C})$ such that $\mathcal{D}_{\hat{C}}(I) > 0$ it holds that $\hat{C}|_I = C$. Then $\hat{C}$ is a $(q, \alpha \cdot \epsilon)$-COLTC.*

A.1.2 Star Products

Now we provide a definition of star products from [43]. These products of codes are very similar to ones used in [36, Section 4], although there exist minor differences. We notice that in this work we are interested mainly in such products of second power (and will not consider m-wise products for $m \geq 3$ as in [43]).

Informally, 2-star product of $C \in \mathbb{F}^n$ is defined by taking tensor product on its core coordinates $A(C)$ and appending its non-core parts (residues) $[n] \setminus A(C)$ to every row/column of such tensor product. This tensor product becomes a core of the obtained start product, and all appended residues are the residue of the star product.

▶ **Definition 10** (Star Products - $C^{\star 2}$). . Let $C \subseteq \mathbb{F}^n$ be a linear code with a γ-core $A(C)$, where $\gamma > 0$ is a constant.

We let $C^{\star 2}$ to be a linear code over $\mathbb{F}$ such that its core and residue will be defined as follows.

we let the core coordinates $A(C^{\star 2}) = A(C) \times A(C)$, and the projection of the code on the core coordinates is $C^{\star 2}|_{A(C^{\star 2})} = C|_{A(C)} \otimes C|_{A(C)}$. The residue of $C^{\star 2}$ (residue(C$^{\star 2}$)) is defined by the residue of every codeword $c \in C^{\star 2}$ as follows:

view every such codeword $c|_{A(C^{\star 2})} \in C^{\star 2}|_{A(C^{\star 2})}$ as a matrix of size $|A(C)| \times |A(C)|$. Its residue(c) is defined by appending for every row r of $c|_{A(C^{\star 2})}$ attach residue residue(r) such that both parts together is a codeword in C, i.e., $(r, \text{residue(r)}) \in$ C. Since $A(C)$ is a core, the residue(r) is defined uniquely given r.

The blocklength of $C^{\star 2}$ is $|A(C)|^2 + 2 \cdot A(C) \cdot (n - A(C))$.

Now, let us define formally the tester for the star product.

▶ **Definition 11** (Tester for Star Product). Let $\mathcal{D}_C$ be a tester for a linear code $C \in \mathbb{F}^n$ with core $A(C)$. Then, a tester for $C^{\star 2}$, denoted by $\mathcal{D}_{C^{\star 2}}$, is defined by:

- pick random $r \in \{0, 1\}$
  - if $r = 0$ pick random row of $A(C^{\star 2})$ and its residue.
  - else pick random column of $A(C^{\star 2})$ and its residue.

Notice that for every local view $I \subseteq \mathrm{coord}(C^{\star 2})$ such that $\mathcal{D}_{C^{\star 2}}(I) > 0$ it holds that $C^{\star 2}|_I = C$. I.e., a local view of such a tester on any codeword of $C^{\star 2}$ is always a codeword of C.

The rest of the proof is organized as follows. A first part of the proof (Section A.2) argues that codes of [11] can be viewed as tensoring over sufficiently large field, together with the *explicit* projection step. The purpose of this part is to define the required abstraction over the codes of [11] needed for our work.

A second part is presented in Section A.3, where it is explained that the codes of Ben-Sasson and Sudan [11] are COLTCs, however, their parameters range is not sufficiently nice. Here we use the fact that their construction can be viewed as a repetitive tensoring as was explained in Section A.2.

Finally, Section A.4 recalls the notion of a relaxed LTC and some related results [42]. It shows that a work of Dinur [18] can be used to improve that soundness of the relaxed LTCs. It turns out that this improvement is sufficient to change the codes of [11] to relaxed LTCs with good enough parameters (which is what we need by Corollary 21). Since the gap amplification technique is explicit, it yields explicit construction of relaxed LTCs. Corollary 21 proves that relaxed LTCs with sufficiently nice parameters can be converted to strong LTCs with constant soundness. That proves Theorem 4.

A.2 Abstraction over the codes of [11]

Let us recall an auxiliary procedure that will be used in the code constructions.

▶ **Definition 12** (Projection of core symbols)**.** Given a linear code $C \subseteq \mathbb{F}^n$ with a code $A(C)$, the projection(C) $\subseteq \mathbb{F}^{\mathrm{n}}$ is a linear code with a new core $A' = A'(C) \subseteq A(C)$, i.e., some core symbols moved to the non-core part.

The abstraction assumed in this Section described in [36, Section 7.2]. For the sake of completeness we give the required details in this section.

Let $\delta, \gamma > 0$ be constants and $\mathbb{F}$ be sufficiently large field. [5]

The explicit construction of [11] can be viewed in the following way. It started from an error-correcting code of constant size blocklength $C_1 \subseteq \mathbb{F}^{n_1}$ such that $A(C_1) = [n_1]$, and for $i = 2 \dots \log\log n$ the following holds.

- $T_i = C_{i-1}^{\star 2}$, i.e., tensoring is done over the core coordinates and residue coordinates are just appended to every row/column code.
- $C_{i+1} = \mathrm{projection}(\mathrm{T_i})$, where a constant fraction of the code coordinates are moved from the "core" part ($A(T_i)$) to the "residue" part (residue($\mathrm{T_i}$)). Notice that projection is the explicitly defined procedure.
- $A(C_{i+1})$ is a γ-core of C_{i+1}, and in particular, $\delta(C_i|_{A(\mathrm{projection}(\mathrm{C_i}))}) \geq \gamma$.
- $\dim(C_{i+1}) = \dim(C_i)^2$
- It is known that $\mathrm{rate}(C_i|_{A(\mathrm{projection}(\mathrm{C_i}))}) \geq \delta$
- It is known that $\mathrm{rate}(C_{i+1}) \geq \delta \cdot \mathrm{rate}(C_i)$

It was proved in [11] that if $w \in \mathbb{F}^{n_i}$ and $w|_{A(C_i)} \notin C_i|_{A(C_i)}$ it holds that the local view of the tester $\mathcal{D}_{C_i}$ is far from the corresponding code, and this distance is proportional to $\delta(w|_{A(C_i)}, C_i|_{A(C_i)})$.

▶ **Claim 13** ([11])**.** *Let i be the number of iteration in the code construction of [11]. Let $C_i \subseteq \mathbb{F}^{n_i}$ be the [11] code from i^{th} iteration and $A(C_i)$ its γ-core for a constant $\gamma > 0$. There exists a constant ϵ_γ depending only on γ such that for every $M \in \mathbb{F}^{|A(C_i)| \times |A(C^i)|}$ the random row/column of M is $\epsilon \cdot \delta(M, C_i|_{A(C_i)} \otimes C_i|_{A(C_i)})$ close to $C_i|_{A(C_i)}$.*

In the classical PCPP structure [11], i.e., the distance is measured only between the core coordinates, and proof (non-core) coordinates help only to test, without being involved into

[5] The codes of [11] are constructed over fields of size linear in a blocklength of the code. The folklore statement says it is not hard to explicitly convert such strong LTCs to be over a constant size field with similar parameters.

the distance calculations. In our scenario (strong locally testable codes) we need to take the proof coordinates into consideration as well.

It is not hard to verify that the above codes construction has the following properties.

▶ **Claim 14.** *Let $C \subseteq \mathbb{F}^n$ be the code from the above construction and $\mathcal{D}_C$ be its tester. Then,*

- *$\mathcal{D}_C$ is uniform over* $\mathrm{supp}(\mathcal{D}_C)$*, and*
- *for each $i \in [n]$ we have $|N_{\mathcal{D}_C}(i)| \leq O(\mathrm{poly}\log(n))$.*

The proof is omitted due to the space limitations.

A.3 The codes of [11] are COLTCs – Main Technical ingredient

Now we reprove Theorem 15 ([43]) showing that the star products are robustly testable with respect to "core robustness" (see Definition 8). Then one can conclude that if C is a q-query COLTC, then $C^{\star 2}$ is a q-query COLTC.

We are ready to prove that the star product of codes has the core robustness.

▶ **Theorem 15** (Star Products has core robustness)**.** *Let C be a linear code with a γ-core $A(C)$. Assume that $\mathcal{D}$ is the star product tester for the code $C^{\star 2}$. Let ϵ_γ be a constant from Claim 13. Then,*

$$\rho^{\mathcal{D}}_{A(C^{\star 2})}(C^{\star 2}) \geq \frac{\epsilon_\gamma \cdot \gamma}{10}.$$

The proof of Theorem 15 is omitted due to the space limitation.

We are ready to claim that the codes of [11] are COLTCs with constant query complexity and inverse poly-log soundness. The proof of the corollary is omitted.

▶ **Corollary 16** (The codes of [11] are COLTCs)**.** *Let $C \subseteq \mathbb{F}^n$ be obtained from the construction in Section A.2 after $O(\log\log n)$ iterations. Then, C is a $(q, \frac{1}{\mathrm{poly}\log n})$-COLTC and as a consequence a $(q, \frac{1}{\mathrm{poly}\log n})$-strong LTC. Moreover, by Claim 14 we have*

- *$\mathcal{D}_C$ is uniform over* $\mathrm{supp}(\mathcal{D}_C)$*, and*
- *for each $i \in [n]$ we have $|N_{\mathcal{D}_C}(i)| \leq O(\mathrm{poly}\log(n))$.*

A.4 Relaxed LTCs and the Gap Amplification

First, we recall a notion of *relaxed LTCs* (rLTCs) Definition 17 from [42]. Intuitively, relaxed LTCs have two kind of coordinates: those with good testability and those which worse (but non-trivial) testability (see Definition 17). Then, we state Theorem 19 and its Corollary 20 saying that the gap amplification of [18] when applied on the codes of [11] yields rLTCs with constant first soundness parameter and inverse poly-log second soundness parameter.

Finally, we recall Corollary 21 from [42] saying that such relaxed LTCs can be easily converted to strong LTCs with constant soundness. Hence Theorem 4 follows.

▶ **Definition 17** (Relaxed LTC)**.** A q-query tester $\mathcal{D}$ is a $(q, \epsilon_1, \epsilon_2)$-rLTC tester for a linear code $C \subseteq \mathbb{F}^n$ with a core $A(C)$, if for every $w \in \mathbb{F}^n$ there exists $c \in C$ such that $\mathop{\mathbf{Pr}}_{I \sim \mathcal{D}}[w|_I \notin C|_I] \geq \max\left\{\epsilon_1 \cdot \delta(w|_{A(C)}, c|_{A(C)}), \epsilon_2 \cdot \delta(w|_{-A(C)}, c|_{-A(C)})\right\}$. A code $C \subseteq \mathbb{F}^n$ with a core $A(C)$ is a $(q, \epsilon_1, \epsilon_2)$-*rLTC* if it has a $(q, \epsilon_1, \epsilon_2)$-rLTC tester.

The parameter q is called the query complexity, ϵ_1 is called the first soundness parameter and ϵ_2 is called the second soundness parameter.

Intuitively, think that ϵ_1 is a constant, but ϵ_2 is sub-constant.

The following simple observation [42] says that any strong LTC is also a relaxed LTC with similar parameters.

▶ **Observation 18** (Strong LTCs are relaxed). *If $C \subseteq \mathbb{F}^n$ is a (q, ϵ)-strong LTC then it is also a $(q, \epsilon, 1)$-rLTC with regards to the code $A(C) = [n]$.*

The observation follows immediately from the definition of relaxed LTCs (Definition 17).

In [42] it was explained that the gap amplification of Dinur can be applied to strong LTCs with inverse poly-log soundness, which can be considered as relaxed LTCs (Observation 18). In this case, the gap amplification preserves it to be an rLTC where first soundness parameter increased by a constant, while the second parameter decreased by a constant. In particular, the arguments of Section 4 and Section 5 in [42] can be summarized to the following theorem describing the affect of the gap amplification technique when applied to the codes from Section A.2.

▶ **Theorem 19** (Stated in [42], Section 5). *Let $q >= 2$ be a constant. Assume $C \subseteq \mathbb{F}^n$ is a $(q, \frac{1}{\text{polylog}(n)})$-strong LTC and $\mathcal{D}_C$ its tester such that it holds that*

- $\text{rate}(C) = \frac{1}{\text{polylog}(n)}$ *and* $\delta(C) \geq \Omega(1)$
- $\mathcal{D}_C$ *is uniform over* $\text{supp}(\mathcal{D}_C)$ *and for each* $i \in [n]$ *we have* $|N_{\mathcal{D}_C}(i)| \leq O(\log n)$.

Then the following holds.

For constant $q \geq 2, \epsilon > 0$, a fixed field $\mathbb{F}$ and infinitely many $n \in \mathbb{N}^+$ we have explicit construction for a linear code $C' \subseteq \mathbb{F}^n$ with a core $A(C')$ such that C' is a $(q, \epsilon, \frac{1}{\text{polylog}(n)})$-rLTC, $\delta(C'|_{A(C')}) = \Omega(1)$ and $\text{rate}(C') = \frac{1}{\text{polylog}(n)}$.

Since Corollary 16 satisfies the assumption of Theorem 19, we conclude the following corollaries.

▶ **Corollary 20.** *For constant $q \geq 2, \epsilon > 0$, a fixed field $\mathbb{F}$ and infinitely many $n \in \mathbb{N}^+$ we have explicit construction for a linear code $C \subseteq \mathbb{F}^n$ with a core $A(C)$ such that C is a $(q, \epsilon, \frac{1}{\text{polylog}(n)})$-rLTC, $\delta(C|_{A(C)}) = \Omega(1)$ and $\text{rate}(C) = \frac{1}{\text{polylog}(n)}$.*

▶ **Corollary 21.** *Assume that for constants $q \geq 2, \epsilon > 0$, field $\mathbb{F}$ and infinitely many $n \in \mathbb{N}^+$ we have a linear code $C \subseteq \mathbb{F}^n$ with a core $A(C)$ such that C is a $(q, \epsilon, \frac{1}{\text{polylog}(n)})$-rLTC, $\delta(C|_{A(C)}) = \Omega(1)$ and $\text{rate}(C) = \frac{1}{\text{polylog}(n)}$. Then, there exists $C' \subseteq \mathbb{F}^{n'}$ such that $n \leq n' \leq n \cdot \text{polylog}(n)$, C' is a $(q, \epsilon/6)$-strong LTC, $\delta(C') = \Omega(1)$ and $\text{rate}(C') = \frac{1}{\text{polylog}(n')}$. Moreover, C' is constructed explicitly from C.*

Theorem 4 follows from Corollary 20 and Corollary 21.